GreenTPU: Predictive Design Paradigm for Improving Timing Error Resilience of a Near-Threshold Tensor Processing Unit

Pramesh Pandey, Prabal Basu, Koushik Chakraborty, and Sanghamitra Roy

***Abstract*— The emergence of hardware accelerators has brought about several orders of magnitude improvement in the speed of the deep neural-network (DNN) inference. Among such DNN accelerators, the Google tensor processing unit (TPU) has transpired to be the best-in-class, offering more than 15× speedup over the contemporary GPUs. However, the rapid growth in several DNN workloads conspires to escalate the energy consumptions of the TPU-based data-centers. In order to restrict the energy consumption of TPUs, we propose GreenTPU—a low-power near-threshold (NTC) TPU design paradigm. To ensure a high inference accuracy at a low-voltage operation, GreenTPU identifies the patterns in the error-causing activation sequences in the systolic array, and prevents further timing errors from similar patterns by intermittently boosting the operating voltage of the specific multiplier-and-accumulator units in the TPU. Compared to a cutting-edge timing error mitigation technique for TPUs, GreenTPU enables 2× to 3× higher performance (TOPS) in an NTC TPU, with a minimal loss in the prediction accuracy.**

***Index Terms*— Deep neural-network (DNN), near-threshold computing (NTC), neural network, predictive, tensor processing unit (TPU), timing error resilience.**

I. Introduction

THE cessation of Dennard's scaling, accompanied by the diminishing throughput from the growing number of on-chip cores, has led to the adoption of power-efficient domain-specific architectures. With the recent confluence of artificial intelligence (AI) and high-performance computing, the domain-specific computing paradigm is already on the uprise, as evident by the success of the deep neural-network (DNN) accelerators [6], [11], [17], [26]. Among the multitude of such *ad-hoc* AI architectures, the Google tensor processing unit (TPU) is at the forefront, claiming 15× to 30× faster inference, compared to the top of the line CPUs and GPUs [13]. However, the unprecedented growth in the DNN workloads (e.g., speech recognition in Google Assistant [2], [13]) portends a rapid increase in the overall power consumption of the Google data-centers. With a view to heavily curtailing the power consumption while sustaining a high inference accuracy, we envision a near-threshold (NTC) operation of the TPUs. However, operating a TPU at the NTC condition can significantly dwindle the inference accuracy due to a high rate of timing errors [9], [34]. This article aims to exploit the inherent architectural artifacts of the TPUs, to predict and tackle the timing errors at NTC, thus promoting a reliable and energy-efficient, low-power TPU design paradigm.

The high delay sensitivity to voltage and process variation (PV) at NTC necessitates a relaxed clock constraint to ensure an error-free execution. On the other hand, hardware accelerators like TPUs are designed to offer a high throughput in niche applications. So, in order to embrace the NTC design paradigm for TPUs, we need to adopt a better-than-worst case design strategy that can efficiently tolerate the timing errors in its systolic array architecture (Section II-A). Prior research efforts delve into the challenges and solutions of tackling timing errors in conventional CPU and contemporary TPU architectures [10], [34]. Next, we discuss why such existing techniques are not effective in an NTC TPU.

Razor—one of the most popular timing error detection and recovery schemes—employs a double sampling flip-flop to detect timing violations inside a pipeline stage [10]. The erring instruction is replayed at a reduced clock frequency to prevent a subsequent timing error. Adopting Razor in TPUs will negatively impact the performance, as the global timing error rate rapidly grows with the dimension of the systolic array. Hence, any recovery penalty, associated with correcting an erroneous computation, will significantly bloat the execution time of the inference. Zhang *et al.* [34] have recently proposed TE-Drop (TD)—where an erring multiplier-and-accumulator (MAC) in a TPU steals a clock cycle from its downstream MAC to correct the error and bypasses the downstream MAC's update. However, this approach cannot tackle any timing error in the last row of MACs, without incurring a significant performance penalty at NTC. As the partial sums grow toward the bottom of the systolic array, the impact of timing errors in the last row of MACs is the most crucial. Moreover, as the rate of the timing error increases significantly at NTC, bypassing the update of some MACs will greatly diminish the inference accuracy.

In the light of such shortcomings of the existing timing error mitigation techniques, we propose a novel *timing error prediction strategy*, exploiting the wavefront propagation of the data in a TPU systolic array. We observe that only a few activation

Manuscript received October 6, 2019; revised February 29, 2020; accepted March 17, 2020. This work was supported by the National Science Foundation (NSF) under Grant CAREER-1253024, Grant CCF-1318826, Grant CNS-1421022, and Grant CNS-1421068. *(Corresponding author: Pramesh Pandey.)*

The authors are with the USU BRIDGE Laboratory, Department of Electrical and Computer Engineering, Utah State University, Logan, UT 84322 USA (e-mail: pandey.pramesh1@aggiemail.usu.edu; prabalb@aggiemail.usu.edu; koushik.chakraborty@usu.edu; sanghamitra.roy@usu.edu).

Color versions of one or more of the figures in this article are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TVLSI.2020.2985057

sequences are more likely to cause timing violations in the MACs (Section II-C). As the activation data streams through all the MACs in a row, an error-causing activation sequence can serve as an excellent predictor to avoid subsequent errors in the rest of the MACs in the same row. We combine this early error prediction scheme with a low-complexity voltage boosting mechanism to propose GreenTPU—a new frontier in the design of reliable and low-power TPU. Following are the key contributions of our work.

1) We observe that only a few input data sequences cause timing violations in MACs. Consequently, they serve as an efficient predictor for impending timing errors (Section II).
2) We propose a heuristic to group several input sequences with similar delay characteristics into a family in order to predict future timing errors in a hardware-efficient manner (Section III-B).
3) We propose GreenTPU—a low-overhead NTC TPU design paradigm that predicts impending timing violations in its systolic array, and precludes them using a novel voltage boosting mechanism (Section III).
4) Combining with our in-house statistical timing analyzer tool, we develop a TPU systolic array simulator in C++. We support an end-to-end integration, by interfacing our simulator with Keras [8], so as to closely emulate a real-life TPU-accelerated inference ecosystem for contemporary DNN applications (Section IV).
5) We demonstrate that GreenTPU provides two orders of magnitude reduction in timing errors at NTC, with respect to TD [34]—a cutting-edge timing error mitigation technique for TPUs (Section V).
6) Compared to TD, GreenTPU offers **2× to 3×** higher performance (TOPS) in an NTC TPU, in seven out of eight DNN data sets, with only 3% average loss in the inference accuracy. Estimated from synthesis, place, and route of a TPU systolic array RTL, augmented with GreenTPU, we find the area, power, and wirelength overheads to be ∼1.8%, ∼2.2%, and ∼4.1%, respectively (Section V).

II. Motivation

In this section, we demonstrate the opportunity of employing a predictive mechanism to tackle timing errors in NTC TPUs. Section II-A provides a background on the TPU systolic array. Using a cross-layer methodology (Section II-B), we analyze the data-driven delay variance in the systolic array of MAC units (Section II-C) and motivate the need for a timing error prediction scheme in NTC TPUs (Section II-D).

A. Background

1) TPU Systolic Array: Matrix multiplication is the most expensive operation in the *inference* phase of the DNN applications. The usage of the systolic array of MAC units has been recognized as a promising direction to accelerate the matrix multiplication. TPU—a DNN accelerator—employs a 256×256 systolic array of MAC units, to multiply the weight matrix with the activation (also referred to as *input*) matrix, maintaining a precision of 8-bit integer [13]. The weights are preloaded into the MACs. The activations stream from the left to the right columns of the array at successive clock cycles. The partial sums from the rows of MACs move downstream. Unlike CPUs and GPUs, *a TPU boasts a distinctly homogeneous architecture with a highly predictable data-flow pattern*.

2) Hazards and Opportunities of NTC TPUs: Operating a TPU at the NTC condition ideally contributes to a quadratic saving in energy consumption. However, the performance of the TPU heavily declines due to a large delay experienced by the circuits at an NTC voltage [9]. Moreover, a high delay sensitivity to PV and voltage variation at NTC demand the clock frequency to be heavily relaxed, compared to a super-threshold operation. Hence, in order to operate with an aggressive clock constraint at NTC, a TPU needs to efficiently tolerate a high rate of timing violations. Furthermore, due to a very deep pipelined architecture of the systolic array (Section II-A1), even a small rate of timing error aids to a severe drop in the inference accuracy of the DNN applications [34].

Fortunately, the architectural homogeneity and a predictable data-flow pattern in TPUs offer a unique opportunity to efficiently tackle timing errors at NTC. Due to a fixed 8-bit precision in the arithmetic operations, we get a finite state space of different sensitized path delays, experienced by the MAC units. Isolating the subset of the relatively high delays, and correlating that subset with the concerned data patterns, can facilitate the prediction of the impending timing errors in the TPU systolic array.

B. Methodology

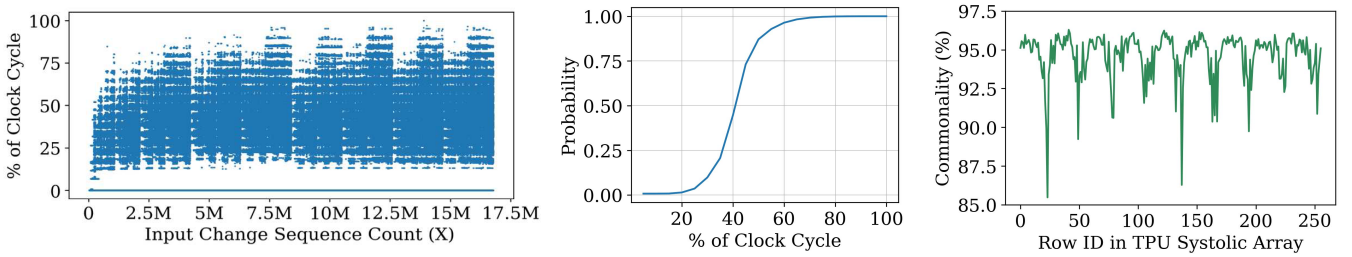
We synthesize a MAC unit at an NTC operating condition (Section V), by using the 15-nm FinFET library from NanGate [23]. We employ our in-house statistical timing analysis (STA) tool to study the delay distributions of the sensitized paths for different inputs to the MAC unit. For a conservative estimate, we consider PV-induced delays, obtained from VARIUS-NTV [27], in randomly chosen 2% of the gates in the MAC circuit [28]. We further elaborate on our cross-layer methodology in Section IV.

C. Results and Significance

The multiplier block of a MAC unit has a relatively deeper logic depth, compared to the accumulator. Hence, we model the delay distribution of the MAC, as a function of the change in inputs to the multiplier, i.e., the activation sequence, and the weight. We create an exhaustive set of 8-bit activation sequences for all possible 8-bit weights, leading to a total of 16 777 216 unique combinations.

Fig. 1(a) shows the delay profile of a PV-affected MAC unit at NTC, obtained by providing all the aforementioned combinations of weights and input changes. A value of X in Fig. 1(a) corresponds to a specific input change sequence, for a specific weight W, as expressed by

$$\text{Weight}\,(W) = \left\lfloor \frac{X}{65536} \right\rfloor, \quad S = X \bmod 65536$$
$$\text{Input change} : \left\lfloor \frac{S}{256} \right\rfloor \rightarrow (S \bmod 256). \tag{1}$$



(a) Exhaustive delay profile of a MAC unit. (b) CDF of the delay profile. (c) Commonality for all the 256 rows.

Fig. 1. (a) Plot of the sensitization delays for all possible weights and input changes for a MAC unit. The variance in the input data can bring about ample delay variance. However, there are only a few input sequences that can sensitize the longest delay paths, as depicted by (b) the CDF plot. (c) Very high percentage of commonality (2) in the error-causing input sequences for all the rows, during the inference of the MNIST data set.

The delay profile shows ample variation, resulting from the variance in the input data. This delay variation is statistically shown as a CDF plot of the delay values in Fig. 1(b), where we conservatively attribute the maximum delay to be the clock period. The key observations from Fig. 1(a) and (b) are: (a) no paths are sensitized when the same activation sequence is applied in two consecutive cycles and (b) a majority of the multiplication operations sensitize paths with low delays. For instance, we notice that the set of delays with more than 60% of the clock cycle is only 3.6% of the entire state space of delays. This sparse sensitization of the higher delay paths eases the prediction of the recurring timing errors from the same input sequence. Next, we discuss the insight into our proposed design of GreenTPU.

D. Timing Error Prediction in TPUs

We aim to systematically study the likelihood of an error-causing input sequence in a MAC, to produce timing errors in the subsequent MACs, belonging to the same row. In this pursuit, we propose a commonality metric as follows:

$$\text{Commonality}_i(\%) = 100*\left(1 - \frac{\bigcup_{j=0}^{255} \text{UES}_j}{\sum_{j=0}^{255} \text{UES}_j}\right) \qquad (2)$$

where UES_j is the set of unique input sequences that cause timing errors in the jth MAC unit of the ith row.

Fig. 1(c) shows a plot of the commonality(%) measured across all the 256 rows during the inference of 1000 test inputs of the MNIST data set. We observe that, for all the rows, the commonality of the error-causing input sequences is more than 85%. This result indicates a *landslide effect* of timing errors in the systolic array of a TPU. In other words, if an input sequence causes a timing error in a MAC unit, that sequence is very likely to cause timing errors in the subsequent MACs, until the sequence is alive in the row. Hence, predicting errors based on the input sequences and adopting a row-wise control strategy can greatly reduce the number of timing errors in a TPU. With this insight, we next discuss GreenTPU—our proposed energy-efficient TPU systolic array design—for a near-threshold operation.

III. GreenTPU

GreenTPU is a novel low-power TPU design paradigm, which dynamically predicts and tackles timing errors in the systolic array of MAC units. Section III-A outlines the design overview. The details of the components of GreenTPU are elaborated in Section III-B through III-E.

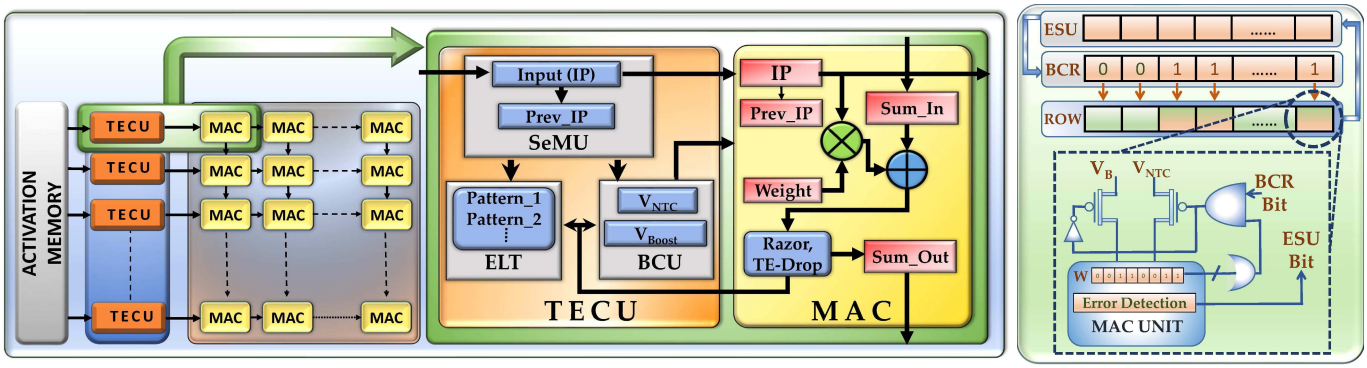
A. Design Overview

Fig. 2(a) shows the top-level design overview of GreenTPU. The heart of GreenTPU is the timing error control unit (TECU). TECU is responsible for predicting and preventing timing errors in the MAC units. In order to maintain a low-complexity circuit design while incurring a negligible performance overhead, we dedicate one TECU per row of MACs, pipelined between the activation memory and the systolic array. A TECU has three main components, namely, error log table (ELT), sequence monitor unit (SeMU), and boost control unit (BCU). When a timing error occurs in any MAC unit of a row, the ELT logs the timing error-causing input sequence pattern. Simultaneously, the BCU is alerted to boost the operating voltage of the subsequent MACs in the row, in order to prevent any future timing error. The SeMU monitors the sequence of inputs and tries to find a matching family representing the pattern in the ELT in every clock cycle. If a match is found, SeMU communicates with the BCU to preclude future timing errors in all the MAC units of a row.

B. Heuristic for Determining Input Sequence Family

As timing error prediction lies at the heart of GreenTPU design, we formulate an efficient heuristic for storing and matching the input sequences responsible for producing higher delays.

We observe that the input sequences with similar delay characteristics can be grouped into a family. The input sequences within a family have similar characteristics of bit flips among them, which are responsible to produce delays that are close to each other. Storing high-delay-causing sequences as families rather than storing each sequence as a different entry is thus a hardware efficient strategy to realize our prediction-based design.



(a) A conceptual block diagram of GreenTPU. (b) BCU and MACs interaction.

Fig. 2. (a) TECUs are pipelined between the activation memory and the rows of the systolic array of MACs. A timing error inside a MAC unit is detected and tackled using Razor and TD techniques, respectively. A TECU comprises an ELT, an SeMU, and a BCU. ELT stores the error-causing input patterns. SeMU, on the other hand, monitors the input data stream and queries the ELT, to identify potential error-causing input sequences. (b) BCU, comprising two 256-bit registers—ESU and BCR, prevents future timing errors by boosting the operating voltage of the MACs in a row.

We analyze the correlation between the input sequences and the delays to group several input sequences into families. We divide the changes of bits in an input sequence into three different groups of bit changes with their respective contribution in producing specific delay or its vicinity: 1) dynamic bit positions, which have the highest domination and are required to flip; 2) static bits positions, which are required to remain static and not flip; and 3) insignificant bits positions, whose flipping is insignificant. One input sequence can thus virtually represent a family of numerous input sequences that produce similar delays by virtue of different combinations of bits in dynamic and insignificant bit positions.

Algorithm 1 methodizes the hardware inexpensive heuristic to store and match the input sequences. In our heuristic, storage of a new input sequence as a family is done by its bit-wise XOR, which can inherently reflect dynamic and static bits positions. Matching is the process of determining if there exists a family in the stored entries, which can encapsulate the input sequence under consideration. Search and incorporation of the insignificant bits are done by loosening the static bit positions by a threshold (line 1 in Algorithm 1). Then, we maintain the domination of dynamic bit positions in the contribution toward the delay (line 9 of Algorithm 1). Consequently, a match is declared, if an entry is found whose static bit positions are same in more than a threshold percentage of positions with the input sequence under consideration (lines 6–16 in Algorithm 1). The threshold serves as a tradeoff between the storage efficiency and the grouping efficiency of the timing error-prone input sequences. Lower threshold enables a family to represent higher number of input sequences, but decreases the grouping efficiency, leading to unwanted voltage boosting. Similarly, a higher threshold more accurately groups the error-causing sequences, while storing more family entries.

Algorithm 1 Pattern Storing/Matching Heuristic

```
1:  TH ← pattern_match_threshold
2:  procedure STORE(current_activ, previous_activ)
3:      xor_pattern ← current_activ ⊕ previous_activ
4:      Store(xor_pattern)
5:  end procedure
6:  procedure MATCH(current_activ, previous_activ)
7:      new_pattern ← current_activ ⊕ previous_activ
8:      for all saved_pat ∈ saved_patterns do
9:          similarity ← saved_pat | new_pattern
10:         num_zeros_sim ← num_reset_bits(saved_pat)
11:         num_zeros_new ← num_reset_bits(similarity)
12:         if num_zeros_new > ⌊TH × num_zeros_sim⌋ then
13:             return match_found
14:         end if
15:     end for
16: end procedure
```

C. Error Log Table (ELT)

ELT is a lookup table that stores the patterns of the input sequence which leads to timing errors in a MAC unit. A timing error in each MAC unit is sensed using a double-sampling flip-flop at the output, similar to Razor [10]. We prevent an erroneous computation from the timing error by employing TD [34], where the errant MAC steals a clock cycle from its downstream MAC to correctly finish its own update. We augment each MAC unit with the capability to store the previous clock cycle's activation input, thus enabling it to infer the input sequence responsible for the timing error. The sequence is then stored as an 8-bit family, as per the STORE procedure of Algorithm 1 (lines 2–4) in the ELT, while the correct output is being computed parallelly. Also, the BCU is signaled with the errant MAC unit's position in the row, to prevent further timing errors in the MAC units, located to the right of the errant MAC. The ELT is implemented as a content addressable memory that enables a fast lookup. When the ELT is full, a pseudo-LRU-based eviction policy is used (not shown in Algorithm 1) to replace an existing pattern with the new incoming pattern. The size of the ELT is a tradeoff

between the hardware overhead and prediction accuracy, which is discussed in Section V.

D. Sequence Monitor Unit (SeMU)

SeMU identifies the possibility of a recurring timing error. The input activation data, coming to each row, are intercepted by SeMU, as the TECU is placed in a pipeline between the activation memory and the systolic array. For a given activation sequence coming from the activation memory, SeMU checks where a corresponding family for that sequence is already present in the ELT, as per the MATCH procedure of Algorithm 1 (lines 6–16). If a match is found, the BCU is alerted to boost the operating voltage of some of the MACs in the row (Section III-E). This action is taken in order to prevent the timing errors that would have been caused by the input sequence. Due to its pipelined architecture, SeMU adds a negligible performance overhead.

E. Boost Control Unit (BCU)

BCU is responsible for boosting the operating voltage of the MAC units, in order to prevent timing errors. As shown in Fig. 2(b), a BCU houses two 256-bit registers: boost control register (BCR) and error sensing unit (ESU). Each bit of these registers corresponds to each MAC unit in a row. Timing error in a MAC is reflected by the setting of the corresponding bit in ESU. We adopt the boosting technique proposed in [22], where every MAC unit has access to two voltage rails, V_{NTC} and V_B, representing a near-threshold and a boost voltage, respectively. The reset (set) value in any bit of the BCR indicates the corresponding MAC unit to operate with the V_{NTC} (V_B) voltage. In our experiments, we set V_{NTC} and V_B to 0.45 and 0.65 V, respectively. Employing the transition infrastructure of [22], we notice that the switching between V_{NTC} and V_B can be performed within one clock cycle of the NTC TPU. Also, we observe that if the preloaded weight of a MAC unit is zero, it is unlikely to encounter a timing error. Hence, a MAC with weight zero can disable the voltage boost for itself, to conserve energy.

The boost control procedures are illustrated in Algorithm 2. Whenever a timing error occurs in any MAC unit ($\text{ESU}_{\text{mac\_}i}$), a certain number of bits of BCR [indicated as resolution (line 1)] that are located to the right of that position are periodically set (line 11) and unset (line 9) as per BOOST_REACTIVE procedure (line 4). One bit in the resolution also reflects one clock cycle to wait for the next boost period (line 7). As a result, the MAC units, specific to those set bits in the BCR, will be boosted in the subsequent cycles, precluding any probable timing violations.

On the other hand, if the SeMU (Section III-D) sends a signal (semu_signal) as a result of finding an errant pattern, BCU starts to periodically boost the entire row, starting from column 0. Boosting of the MAC units will happen periodically in response to the set bits in BCR. Again, a certain number of BCR bits are set at a time, defined by resolution as per the BOOST_PROACTIVE procedure (line 15). The resolution is empirically ascertained. The choice is guided by the energy budget of the GreenTPU implementation and the noise margin of the TPU systolic array at NTC to tradeoff between the tolerable timing errors and the energy overhead.

Algorithm 2 BCU Algorithm

```
1:  resolution ← no_of_bits_to_set
2:  BCR ← Boost_Control_Register
3:  ESU ← Error_Sensing_Unit
4:  procedure BOOST_REACTIVE(ESU_mac_i)
5:      start ← mac_i + 1
6:      while start < 256 do
7:          wait_for_clock_cycles(resolution)
8:          if start ≠ mac_i then
9:              BCR[start − resolution ... start − 1] ← 0
10:         end if
11:         BCR[start ... min(start + resolution − 1, 255)] ← 1
12:         start ← start + resolution
13:     end while
14: end procedure
15: procedure BOOST_PROACTIVE(semu_signal)
16:     start ← 0
17:     while start < 256 do
18:         wait_for_clock_cycles(resolution)
19:         if start ≠ 0 then
20:             BCR[start − resolution ... start − 1] ← 0
21:         end if
22:         BCR[start ... start + resolution − 1] ← 1
23:         start ← start + resolution
24:     end while
25: end procedure
```

F. GreenTPU Variants

We outline three different variants of GreenTPU to better understand the implication of different architectural artifacts of GreenTPU design.

1) GreenTPU: GreenTPU is the variant, which includes every detail of the architecture discussed so far. It includes a full-fledge predictive engine, which can store many error-causing input sequence families to facilitate prediction of any imminent timing errors from the input sequences represented by those families, as defined by the pattern-match threshold in Algorithm 1. This variant helps us to better understand the efficacy of a full-blown predictive approach for maintaining the DNN accuracy in a high-timing error-prone environment.

2) GreenTPU Reactive (GTR): GreenTPU Reactive (GTR) is a variant without any form of predictive capabilities of GreenTPU design paradigm. When the timing error in a MAC unit is detected as a result of an error-causing sequence, the MAC units to the right in the row are prevented from potential timing errors from the same sequence. Architecturally, GTR omits SeMU and ELT altogether and only uses BOOST_REACTIVE procedure (line 3 of Algorithm 2) for BCU. GTR helps to better understand the extent of efficacy than can be provided by a purely reactive approach in maintaining DNN accuracy in a high-timing error-prone environment.

3) GreenTPU Lite (GTL): GreenTPU Lite (GTL) is a variant which introduces a hint of predictiveness over GTR. It does so by including the minimum possible predictive engine for the most basic prediction scheme. Only one error-causing input sequence is used as the basis of prediction and that entry is constantly replaced on the introduction of a new timing error. Hence, architecturally, the entire ELT is replaced by an 8-bit register to store the XOR of the error-causing sequence. The pattern-match threshold in Algorithm 1 is set to 100% to simplify the prediction scheme, thereby flagging a match only upon an exact match between stored XOR pattern and the XOR pattern of the incoming input sequence. GTL helps to better understand the efficacy added by the basic introduction of predictiveness over a reactive timing error correction scheme for DNN accelerators.

IV. Methodology

In this section, we explain our extensive cross-layer methodology, which is used to implement and evaluate GreenTPU variants.

A. Device Layer

We estimate the NTC energy consumptions by performing HSPICE simulations on the basic logic gates (namely, NAND, XOR, and Inverter). We use the 31-stage FO4 inverter-chain as a representative of various combinational logics in a TPU. The simulation parameters are obtained from the 16-nm predictive technology model [36]. We incorporate the impact of the PV at NTC using the VARIUS-NTV [15] model. The FinFET characteristics are obtained using the VARIUS-TC model [16]. The delays of the basic gates are used in the circuit layer (Section IV-B) to ascertain the sensitized path delays in a MAC.

B. Circuit Layer

We develop the Verilog RTL description of a systolic array and augment it with the GreenTPU components. We synthesize the RTLs using the Synopsys design compiler, at various operating conditions. We perform place and route of the synthesized netlist using Cadence system-on-chip (SoC) Encounter, and estimate the area, power, and wirelength overheads at the NTC operating condition. Using both synthetically generated, as well as, real data set-driven inputs, we obtain the sensitized path delays in the MAC array with our in-house STA tool. Based on a library of the delay files of the basic logic gates at different operating voltages, the STA tool reports the delays of the sensitized paths of the MAC circuit.

C. Architecture Layer

Based on the architectural description detailed in [13], we develop a cycle-accurate TPU systolic array simulator—TPU-Sim—in C++, and implement the GreenTPU components in TPU-Sim. We integrate the STA tool (Section IV-B) with TPU-Sim, to accurately model timing errors in the MACs, based on real data-driven sensitized path delays. We create a real TPU-based inference ecosystem by conjoining TPU-Sim with Keras [8]. First, we train several DNN applications (namely, MNIST [19], Reuters [3], CIFAR-10 [18], IMDB [21], SVHN [24], GTSRB [29], FMNIST [33], and FSDD [1]) using Keras, running TensorFlow in the back-end. We extract each layer's activation inputs and trained model weights, and preprocess them into multiple 256×256 8-bit-integer matrices. TPU-Sim is invoked with each pair of the preprocessed input and weight matrices. The output matrices from the TPU-Sim are combined to evaluate the inference accuracy. We parallelize our framework for handling a large amount of test data using Python Multiprocessing.

V. Experimental Results

In this section, we evaluate the efficacy of different timing error-resilient schemes, when a TPU operates at a better-than-worst case scenario. Our baseline NTC operating condition (0.45 V, 67 MHz) guarantees an error-free execution of the TPU. Section V-A describes the comparative schemes. Section V-B elaborates on the timing error resilience of the schemes. Section V-C presents the inference accuracy and the energy consumption of the TPU under different schemes. Finally, Section V-D discusses the hardware overheads of GreenTPU.

A. Comparative Schemes

1) *TD:* This is a recently proposed technique that can tackle timing errors in the systolic array of a TPU [34]. The errant MAC steals the next clock cycle from its downstream MAC to correct the error, while the downstream MAC bypasses its own operation.
2) *GreenTPU (GT):* This is our proposed design strategy that stores the error-causing patterns in order to predict any imminent timing errors from those patterns (Section III). This variant has a full-fledge predictive engine in place (Section III-F1). For the experiments, the pattern-match threshold of 90%, and an ELT size of 10 is chosen.
3) *GTL:* This is a lighter variant of GreenTPU, with the most basic predictive engine capable of storing only one error-causing pattern (Section III-F3).
4) *GTR:* This is a variant of GreenTPU with the predictive capability taken off (Section III-F1). The prevention of timing errors thus only occurs reactively after a timing error in the row has occurred.

B. Timing Error Resilience

Fig. 3 depicts the number of timing errors encountered during the inference of the DNN data sets under different schemes, when the TPU operates at a higher frequency, compared to the baseline. However, at all frequency—denoted by the X-axis—the operating voltage is kept constant at 0.45 V. The Y-axis values are represented on a logarithmic scale. We notice that, on an average, GT encounters two orders of magnitude less timing errors, with respect to TD, across all the data sets, at any higher performance level. GT, boasting a full-blown prediction engine, attributes to this huge reduction.

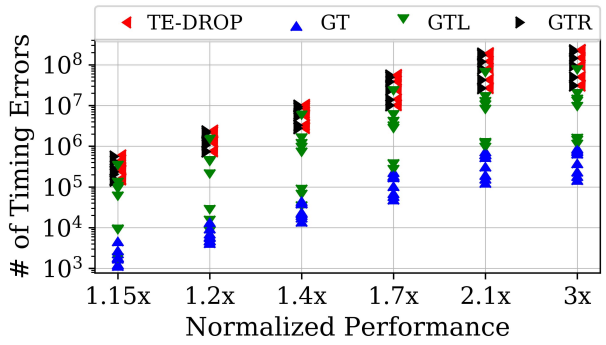


Fig. 3. Number of timing errors encountered in different comparative schemes across eight DNN data sets.

GT is seen as capable of predicting most of the timing errors and preventing them from occurring. Although equipped with a preliminary prediction scheme, GTL is still seen to substantially reduce the number of timing errors compared to TD and GTR. On the other hand, as GTR does not have any form of prediction mechanism, the reduction in the number of timing errors in the log scale is almost negligible. These results demonstrate the importance of predictive approaches when it comes to an exponential reduction of the timing errors. It shows that predictive approaches are the way to go when we have to aggressively scale up the performance of a massively parallel architecture like NTC TPU.

C. Inference Accuracy and Energy

Fig. 4 presents the variations in the inference accuracy at different performance points (Section V-B), under various comparative schemes (Section V-A), for eight DNN data sets. The accuracy values of the data sets are normalized to the corresponding error-free accuracy (IMDB: 0.90, CIFAR-10: 0.77, MNIST: 0.98, REUTERS: 0.80, FSDD: 0.92, FMNIST: 0.89, GTSRB: 0.97, and SVHN: 0.94) from the baseline NTC TPU. Fig. 4 also shows the voltage boosting energy (VBE), associated with the boosting mechanism in GT and GTL. VBE is calculated as a percentage of the energy consumption of the baseline NTC systolic array with no augmentation. We see that the accuracy curves (left Y-axis) fall from the normalized maximum at different rates, due to varied timing error resilience of different schemes. Also, the VBE curves (right Y-axis) rise from the minimum, reflecting the different rates of increase in the number of voltage boosting events necessary to provide the required timing error resilience for different schemes.

Up to 1.4× the baseline performance, all the schemes can efficiently prevent the impact of timing errors from affecting the inference accuracy. However, as the performance is further increased, GT and GTL offer considerably better accuracies with respect to TD and GTR, for all the data sets. This is due to the high timing error resilience of GT and GTL (Section V-B). The pattern matching capability, along with a larger ELT, makes GT a more effective scheme, compared to GTL. Our baseline NTC TPU, augmented with GT, can be operated at

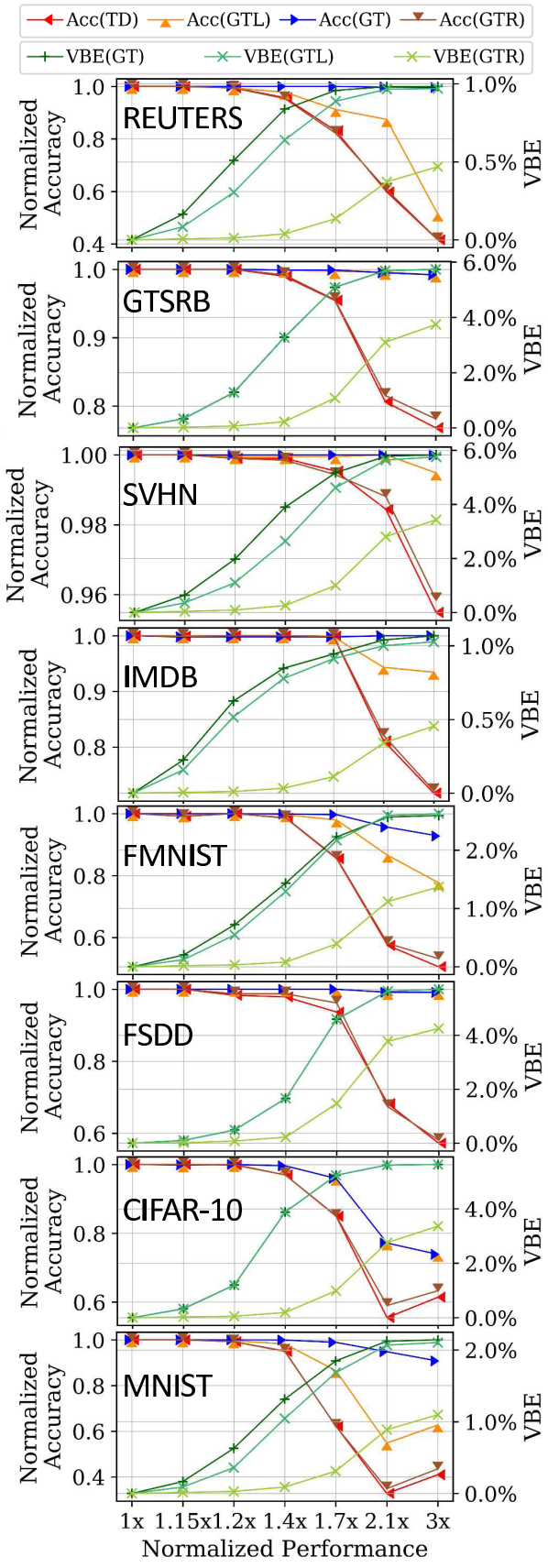


Fig. 4. Normalized inference accuracy (Acc), and VBE from the comparative schemes, at different normalized performance levels, across eight DNN data sets.

2× to 3× the baseline frequency, with only 3% average loss in the inference accuracy for seven out of eight DNN data sets. For CIFAR-10, GT is only as effective as GTL. This anomaly is attributed to the extreme variance in the activation patterns of CIFAR-10. GTR performs better or equal to TD in all the cases; however, a noticeable increase in accuracy relative to TD is not seen. We see that even the most basic prediction scheme added to a reactive approach (GTR to GTL) can have a huge impact on maintaining the inference accuracy. This clearly shows that the maintenance of DNN inference accuracy at aggressively higher performance points can only be achieved by near-exponentially reducing the potential timing errors. Remaining in line with the capability of predictive schemes to be able to exponentially reduce the number of timing errors (Section V-B), predictive schemes GT and GTL maintain the accuracy far better than the nonpredictive and reactive approaches like GTR and TD.

The VBE of GTL—due to its lower hardware footprint and infrequent boosting—is usually less than the VBE of GT. However, for CIFAR-10, both the schemes trigger the boosting mechanism for the same number of times, thus incurring similar energy overheads. GTR incurs the lowest VBE of all the GreenTPU variants as it has to perform boosting for the least number of times, provided that the GTR is not a predictive scheme. Despite a monotonic increase with the performance, VBEs of our proposed schemes are limited to ∼6% of the baseline NTC TPU energy consumption. This result is due to the sporadic occurrence of the boosting. Due to the sparsity of need for boosting, it can be concluded that, rather than selectively undervolting the MACs in a TPU operating at super threshold voltage, it is highly beneficial from an energy perspective, to operate the TPU at the near-threshold voltage and selectively boost the MACs in a TPU. The performance loss coming from this setting can be effectively uplifted by GT to yield a highly energy-efficient TPU. Hence, GT serves as an extremely error-resilient and energy-efficient design paradigm that can unlock a high performance in future low-power NTC TPUs. Furthermore, as GT is based on the hardware level data-delay relationship at the basic granularity of a MAC unit, it can scale well to systolic array dimension, the bit width of activation/weight, and the size of the DNN applications.

D. Implementation Overheads

The hardware overheads of GreenTPU come from the TECU components, the additional voltage rail, and the augmentation of each MAC with the Razor capability. The area overhead of GreenTPU is estimated to be ∼1.8%. This small footprint is attributed to the fact that the systolic array occupies only 24% of the overall TPU die area [13]. GreenTPU incurs a power overhead of ∼2.2%, compared to the vector-less power consumption of the systolic array. From the detailed route reports, GreenTPU's wire-length overhead is estimated to be ∼4.1%.

VI. Related Work

Prior research efforts related to our work deal with improving the energy efficiency of the DNN hardware accelerators through various means. Chen *et al.* [6] proposed an optimal MAC operation mapping rule, called Row-Stationary dataflow, which optimizes the data movement inside a deep convolutional neural network (CNN), resulting in a superior system-level energy efficiency. Reagen *et al.* [26] demonstrated an automated codesign approach across the algorithm, architecture, and circuit to offer a staggering 8.1× power reduction over a baseline DNN accelerator, without compromising the accuracy. Lin *et al.* [20] presented a statistical error compensation technique to correct the process variation-induced timing errors in CNNs, operating under near-threshold condition.

Zhang *et al.* [34] proposed a timing speculation approach that enables an aggressive voltage underscaling in DNN accelerators without compromising the classification accuracy. Choi *et al.* [7] proposed error resilient techniques to enable aggressive voltage scaling by exploiting the variable error resilience exposed by different components of DNN. Zhang *et al.* [35] proposed design of fault-tolerant, systolic array-based DNN accelerators for high defect rate technologies in the case of permanent hardware faults. Chandramoorti *et al.* [5] present a technique of low-voltage neural network acceleration with application-aware SRAM architecture. Nguyen *et al.* [25] innovate in error resilience around DRAM accesses to increase the energy efficiency of DNN applications. Kim *et al.* [17] presented a memory adaptive training with *in-situ* canaries, which enables aggressive voltage scaling of DNN-accelerator weight memories to improve the energy efficiency.

Whatmough *et al.* [4], [32] have incorporated several energy-efficient hardware techniques, such as curbing unwanted computations, providing algorithmic error tolerance, and timing violation tolerance, into their DNN SoC. They provide the timing error tolerance by complementing Razor with time borrowing [30], [31]. Hegde *et al.* [12] propose a predictive scheme to tackle timing errors coming as a result of critical undervolting in DSP architectures. Karakonstanis *et al.* [14] propose a undervolting-enabled DCT architecture to demonstrate higher energy savings. However, our work is the first one that exploits the data-driven delay variance in the systolic array of MACs, to predict timing errors in TPUs, operating under near-threshold condition.

VII. Conclusion

The unprecedented growth of the DNN workloads in the recent years requires an energy-efficient DNN accelerator design paradigm, which can offer an optimal inference accuracy at a high performance. In this article, we present GreenTPU—an energy-optimized systolic array design for Google TPU—a state-of-the-art DNN accelerator. Operating at the NTC condition, GreenTPU can efficiently predict and prevent the imminent timing errors in its systolic array of MACs, thus offering close to an error-free accuracy with a high performance. We also establish that predictive approaches to error resilience have the required potential to maintain DNN inference accuracy in aggressively performance-scaled DNN accelerator platforms. Compared to a recently proposed timing error mitigation strategy for TPUs, GreenTPU enables

2× to 3× higher performance (TOPS) in an NTC TPU, with a minimal loss in the prediction accuracy, and minor hardware footprints.

REFERENCES

[1] *Free Spoken Digit Dataset*. Accessed: May 2019. [Online]. Available: https://github.com/Jakobovski/free-spoken-digit-dataset

[2] O. Google. *Siri, Alexa, Cortana; Can You Tell Me Some Stats on Voice Search*. Accessed: May 2019. [Online]. Available: https://edit.co.uk/blog/google-voice-search-stats-growth-trends/

[3] University of California, Irvine, Irvine, CA, USA. *Reuters-21578 Dataset*. Accessed: May 2019. [Online]. Available: http://kdd.ics.uci.edu/databases/reuters21578/reuters21578.html

[4] P. N. Whatmough, S. K. Lee, H. Lee, S. Rama, D. Brooks, and G.-Y. Wei, "14.3 A 28 nm SoC with a 1.2 GHz 568 nJ/prediction sparse deep-neural-network engine with >0.1 timing error rate tolerance for IoT applications," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, Feb. 2017, pp. 242–243.

[5] N. Chandramoorthy *et al.*, "Resilient low voltage accelerators for high energy efficiency," in *Proc. IEEE Int. Symp. High Perform. Comput. Archit. (HPCA)*, Feb. 2019, pp. 147–158.

[6] Y.-H. Chen, J. Emer, and V. Sze, "Using dataflow to optimize energy efficiency of deep neural network accelerators," *IEEE Micro*, vol. 37, no. 3, pp. 12–21, Apr. 2017.

[7] W. Choi, D. Shin, J. Park, and S. Ghosh, "Sensitivity based error resilient techniques for energy efficient deep neural network accelerators," in *Proc. 56th Annu. Design Autom. Conf. (DAC)*, New York, NY, USA, 2019, p. 204.

[8] F. Chollet. (2015). *Keras*. [Online]. Available: https://keras.io

[9] R. G. Dreslinski, M. Wieckowski, D. Blaauw, D. Sylvester, and T. Mudge, "Near-threshold computing: Reclaiming Moore's law through energy efficient integrated circuits," *Proc. IEEE*, vol. 98, no. 2, pp. 253–266, Feb. 2010.

[10] D. Ernst *et al.*, "Razor: A low-power pipeline based on circuit-level timing speculation," in *Proc. 22nd Digit. Avionics Syst. Conf. Process.*, 2000, pp. 7–18.

[11] V. Gokhale, A. Zaidy, A. X. M. Chang, and E. Culurciello, "Snowflake: An efficient hardware accelerator for convolutional neural networks," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, May 2017, pp. 1–4.

[12] R. Hegde and N. R. Shanbhag, "Soft digital signal processing," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 9, no. 6, pp. 813–823, Dec. 2001.

[13] N. P. Jouppi *et al.*, "In-datacenter performance analysis of a tensor processing unit," in *Proc. ACM/IEEE 44th Annu. Int. Symp. Comput. Archit. (ISCA)*, Jun. 2017, pp. 1–12.

[14] G. Karakonstantis, N. Banerjee, and K. Roy, "Process-variation resilient and voltage-scalable DCT architecture for robust low-power computing," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 18, no. 10, pp. 1461–1470, Oct. 2010.

[15] U. R. Karpuzcu, K. B. Kolluru, N. S. Kim, and J. Torrellas, "VARIUS-NTV: A microarchitectural model to capture the increased sensitivity of manycores to process variations at near-threshold voltages," in *Proc. IEEE/IFIP Int. Conf. Dependable Syst. Netw. (DSN)*, Jun. 2012, pp. 1–11.

[16] S. K. Khatamifard, M. Resch, N. S. Kim, and U. R. Karpuzcu, "VARIUS-TC: A modular architecture-level model of parametric variation for thin-channel switches," in *Proc. IEEE 34th Int. Conf. Comput. Design (ICCD)*, Oct. 2016, pp. 654–661.

[17] S. Kim, P. Howe, T. Moreau, A. Alaghi, L. Ceze, and V. S. Sathe, "Energy-efficient neural network acceleration in the presence of bit-level memory errors," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 65, no. 12, pp. 4285–4298, Dec. 2018.

[18] A. Krizhevsky, "Learning multiple layers of features from tiny images," Univ. Toronto, Toronto, ON, Canada, Tech. Rep. [Online]. Available: http://www.cs.toronto.edu/ kriz/learning-features-2009-TR.pdf

[19] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[20] Y. Lin, S. Zhang, and N. R. Shanbhag, "Variation-tolerant architectures for convolutional neural networks in the near threshold voltage regime," in *Proc. IEEE Int. Workshop Signal Process. Syst. (SiPS)*, Oct. 2016, pp. 17–22.

[21] A. L. Maas, R. E. Daly, P. T. Pham, D. Huang, A. Y. Ng, and C. Potts, "Learning word vectors for sentiment analysis," in *Proc. 49th Annu. Meeting Assoc. Comput. Linguistics, Hum. Lang. Technol.*, 2011, pp. 142–150.

[22] T. N. Miller, X. Pan, R. Thomas, N. Sedaghati, and R. Teodorescu, "Booster: Reactive core acceleration for mitigating the effects of process variation and application imbalance in low-voltage chips," in *Proc. IEEE Int. Symp. High-Perform. Comp Archit.*, Feb. 2012, pp. 1–12.

[23] *NANGATE*. Accessed: May 2019. [Online]. Available: http://www.nangate.com/?page id=2328

[24] Y. Netzer, T. Wang, A. Coates, and A. Bissacco, "Reading digits in natural images with unsupervised feature learning," in *Proc. NIPS Workshop Deep Learn. Unsupervised Feature Learn.*, 2011, pp. 1–9.

[25] D.-T. Nguyen, N.-M. Ho, and I.-J. Chang, "St-DRC: Stretchable DRAM refresh controller with no parity-overhead error correction scheme for energy-efficient DNNs," in *Proc. 56th Annu. Design Autom. Conf. (DAC)*, 2019, p. 205.

[26] B. Reagen *et al.*, "Minerva: Enabling low-power, highly-accurate deep neural network accelerators," in *Proc. ACM/IEEE 43rd Annu. Int. Symp. Comput. Archit. (ISCA)*, Jun. 2016, pp. 267–278.

[27] S. R. Sarangi, B. Greskamp, R. Teodorescu, J. Nakano, A. Tiwari, and J. Torrellas, "VARIUS: A model of process variation and resulting timing errors for microarchitects," *IEEE Trans. Semicond. Manuf.*, vol. 21, no. 1, pp. 3–13, Jun. 2008.

[28] T. Shabanian, A. Bal, P. Basu, K. Chakraborty, and S. Roy, "ACE-GPU: Tackling choke point induced performance bottlenecks in a near-threshold computing GPU," in *Proc. ISLPED*, 2018, pp. 1–6.

[29] J. Stallkamp, M. Schlipsing, J. Salmen, and C. Igel, "Man vs. Computer: Benchmarking machine learning algorithms for traffic sign recognition," *Neural Netw.*, vol. 32, pp. 323–332, Aug. 2012.

[30] P. N. Whatmough, S. Das, and D. M. Bull, "A low-power 1-GHz razor FIR accelerator with time-borrow tracking pipeline and approximate error correction in 65-nm CMOS," *IEEE J. Solid-State Circuits*, vol. 49, no. 1, pp. 84–94, Jan. 2014.

[31] P. N. Whatmough, S. Das, D. M. Bull, and I. Darwazeh, "Circuit-level timing error tolerance for low-power DSP filters and transforms," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 21, no. 6, pp. 989–999, Jun. 2013.

[32] P. N. Whatmough, S. K. Lee, D. Brooks, and G.-Y. Wei, "DNN engine: A 28-nm timing-error tolerant sparse deep neural network processor for IoT applications," *IEEE J. Solid-State Circuits*, vol. 53, no. 9, pp. 2722–2731, Sep. 2018.

[33] H. Xiao, K. Rasul, and R. Vollgraf, "Fashion-MNIST: A novel image dataset for benchmarking machine learning algorithms," *CoRR*, Sep. 2017, arXiv:1708.07747. [Online]. Available: https://arxiv.org/abs/1708.07747.

[34] J. Zhang, K. Rangineni, Z. Ghodsi, and S. Garg, "ThUnderVolt: Enabling aggressive voltage underscaling and timing error resilience for energy efficient deep neural network accelerators," 2018, *arXiv:1802.03806*. [Online]. Available: http://arxiv.org/abs/1802.03806

[35] J. J. Zhang, T. Gu, K. Basu, and S. Garg, "Analyzing and mitigating the impact of permanent faults on a systolic array based neural network accelerator," in *Proc. IEEE 36th VLSI Test Symp. (VTS)*, Apr. 2018, pp. 1–6.

[36] W. Zhao and Y. Cao, "New generation of predictive technology model for sub-45nm early design exploration," *Trans. Electron Devices*, vol. 53, no. 11, pp. 2816 –2823, 2006.

Pramesh Pandey received the B.Tech. degree from NIT Allahabad, Allahabad, India, in 2015. He is currently pursuing the Ph.D. degree at the BRIDGE Laboratory, Electrical and Computer Engineering Department, Utah State University, Logan, UT, USA, under the mentorship and co-mentorship of Dr. Sanghamitra Roy and Dr. Koushik Chakraborty, respectively.

His research interests span around low-power computing and the associated reliability, performance, and security issues. He is recently delving his time into exploring domain-specific architectures at low voltages.

Prabal Basu received the Ph.D. degree from the Electrical and Computer Engineering Department, Utah State University, Logan, UT, USA, in 2019, under the mentorship of Dr. Koushik Chakraborty.

He is currently working as an Electronic Design Automation Engineer with Cadence Design Systems, San Jose, CA, USA. His research expertise are in the areas of reliable network-on-chip design, energy-efficient graphics processing unit circuit-architecture codesign, error-resilient systolic array design for near-threshold voltage operation, and the security of low-power computing.

Koushik Chakraborty received the B.Tech. degree from IIT Kanpur, Kanpur, India, in 2000, and the M.S. and Ph.D. degrees from the University of Wisconsin–Madison, Madison, WI, USA, in 2004 and 2008, respectively.

He is currently an Associate Professor with the Electrical and Computer Engineering Department, Utah State University, Logan, UT, USA. His current research interests are cross-layer circuit-architectural techniques to improve the energy efficiency and reliability of microprocessors. His research is currently funded by National Science Foundation and Micrometer Incorporation.

Dr. Chakraborty received the Best Paper Award at the 2010 VLSID, the 2011 DATE, 2012 ICCD, and the Best Paper Nominations at 2014 CODES-ISSS.

Sanghamitra Roy received the M.S. degree in computer engineering from Northwestern University, Evanston, IL, USA, in December 2003, and the Ph.D. degree from the Department of Electrical and Computer Engineering, University of Wisconsin–Madison, Madison, Madison, WI, USA, in 2008. Her doctoral research was sponsored by Intel Strategic CAD labs and National Science Foundation.

She is currently an Associate Professor with the Department of Electrical and Computer Engineering, Utah State University, Logan, UT, USA. She has authored over 50 peer-reviewed publications in top tier journals and conferences as well as a book chapter in *VLSI Design Automation*. Her research interests are in VLSI circuit design and optimization, and exploring reliability-aware novel circuit styles and architectures.

Dr. Roy received the Best Paper Award nominations at the IEEE/ACM International Conference on Computer Aided Design (ICCAD) in 2005, the IEEE 23rd International Conference on VLSI Design (VLSI Design) in 2010, the IEEE Design Automation and Test in Europe (DATE) in 2011, the Best Paper Award at the 30th IEEE International Conference on Computer Design (ICCD) in 2012, and the NSF CAREER Award in 2013. She serves in the Editorial Board of the IEEE Design and Test Magazine and in the Technical Program Committees of DAC, ICCD, and ISQED.