Tolerating Soft Errors in Deep Learning Accelerators with Reliable On-Chip Memory Designs

Arash Azizimazreah*, Yongbin Gu*, Xiang Gu, Lizhong Chen
School of Electrical Engineering and Computer Science
Oregon State University, Corvallis, USA
{azizimaa, guyo, guxi, chenliz}@oregonstate.edu

***Abstract*—Deep learning neural network (DNN) accelerators have been increasingly deployed in many fields recently, including safety-critical applications such as autonomous vehicles and unmanned aircrafts. Meanwhile, the vulnerability of DNN accelerators to soft errors (e.g., caused by high-energy particle strikes) rapidly increases as manufacturing technology continues to scale down. A failure in the operation of DNN accelerators may lead to catastrophic consequences. Among the existing reliability techniques that can be applied to DNN accelerators, fully-hardened SRAM cells are more attractive due to their low overhead in terms of area, power and delay. However, current fully-hardened SRAM cells can only tolerate soft errors produced by single-node-upsets (SNUs), and cannot fully resist the soft errors caused by multiple-node-upsets (MNUs). In this paper, a Zero-Biased MNU-Aware SRAM Cell (ZBMA) is proposed for DNN accelerators based on two observations: first, the data (feature maps, weights) in DNNs has a strong bias towards zero; second, data flipping from zero to one is more likely to cause a failure of DNN outputs. The proposed memory cell provides a robust immunity against node upsets, and reduces the leakage current dramatically when zero is stored in the cell. Evaluation results show that when the proposed memory cell is integrated in a DNN accelerator, the total static power of the accelerator is reduced by 2.6X and 1.79X compared with the one based on the conventional and on state-of-the-art full-hardened memory cells, respectively. In terms of reliability, the DNN accelerator based on the proposed memory cell can reduce 99.99% of false outputs caused by soft errors across different DNNs.**

I. INTRODUCTION

Deep learning neural networks (DNNs) are popular in many fields such as speech recognition, autonomous vehicles, and data centers as they can achieve unprecedented accuracy. To overcome the computation bottleneck of traditional platforms, researchers proposed specialized DNN accelerators consisting of up to thousands of processing elements to accelerate the inference process, such as Google Tensor processing units (TPUs) [1] and Eyeriss [2].

While DNNs have been increasingly used in many applications, their reliability is rarely investigated. The primary unreliable factor in modern systems mainly comes from the soft errors typically resulted from high-energy particle strikes. These soft errors can change the stored data (e.g. bit flips), which may further cause large deviation of standard system outputs. As silicon devices are scaled down to smaller dimensions for a denser integration, the soft error rate (SER) increases drastically, and has become a major challenge in the reliability of silicon chips at the terrestrial level [3]–[5]. For instance, suppose that a stream of data from cameras goes to a DNN accelerator for object detection and classification in autonomous vehicles [6]. If the autonomous vehicle misclassifies a truck or a pedestrian as a flying object due to soft errors, the vehicle may not execute the brake operation to avoid the collision [6]. Therefore, the reliability investigation becomes imperative especially in some safety-critical fields, as any failure in output may lead to catastrophic consequences.

The power efficiency of the DNN accelerators has also drawn a wide attention when they are deployed in many power-constraint environments, like IoTs [7]. As silicon fabrication technologies are scaled down to smaller dimensions, the static power caused by leakage current accounts for a major part of the total chip power [8]–[10]. Among modern DNN accelerators, a significant part of the chip (e.g. 75%) is used for on-chip buffers. These on-chip buffers are typically implemented by static random access memories (SRAMs). Hence, a large amount of power is consumed by the leakage current of SRAMs. The situation becomes even worst since there is an increasing trend to use compact data representations in DNNs to improve the efficiency, as more chip area will be used to deploy SRAMs. [11]. This trend results in memory-orientated DNN accelerators as computational units of accelerators is simplified due to compact data representations, however, larger on-chip buffer is used to store more data on-chip in order to reduce the off-chip traffic [12]. Thus, beyond considering the reliability of the DNN accelerators, static power of the on-chip memory is also a non-negligible issue when designing power efficient DNN accelerators.

There are several methods (Error Correction Codes, TMR, Fully-hardened SRAM cells) that can be used to tolerate soft errors in DNN accelerators [6]. In terms of area, power and delay, the fully-hardened SRAM cell techniques become a promising way to improve the system reliability [13]. However, existing hardened SRAM cells only have limited capability to tolerate soft errors caused by Multiple-Node-Upsets (MNUs) while the probability of MNUs in SRAM cells increases as the fabrication technologies scale down [3]–[5]. Worse still, compared with a low-leakage SRAM cell, the existing fully-hardened SRAM cells consume quite amount of leakage current in a nano-scaled technology which leads to significant power consumption. In response to these drawbacks, our objective is to design a memory cell for DNN accelerators that can eliminate both singe-node-upsets (SNUs) and MNUs. In the meantime, the proposed cell should have smaller leakage current compared with other fully-hardened

* These authors contributed equally.

978-1-5386-8367-5/18/$31.00 ©2018 IEEE

SRAM cells. The main obstacle in achieving the immunity against MNUs is the positive feedback between storage nodes within the circuitry of memory cells. This positive feedback is used to store the data in the cell, however, it is also the main reason of bit upsets at MNU occurrence in existing memory cells, even if they are fully-hardened against SNU.

Previous researches [2] [14] [15] have revealed that the data (feature maps, weights) in DNNs has a strong bias towards zero. Statistics show that feature maps can have up to 75% zeros [2] [14], while weights can reach 96% [15]. In addition, a recent study [6] and our experiment both have observed that bit flips from zero to one more likely lead to DNN output failures compared with bit flips from one to zero. Based on these observations, a novel low-power fully-hardened memory cell with capability of tolerating MNUs for zero (ZBMA) is proposed for the DNN accelerators. Our simulation results show that a DNN accelerator based on our proposed memory cell can achieve 99.99% protection against soft errors across different DNNs while the static power is reduced by 1.79X compared with current state-of-the-art. The main contributions of this paper are the following:

- Augmenting a light-weight deep learning framework, Tiny-DNN [16], to enable random fault injection in different positions for various networks.
- Analyzing fault propagation of three popular networks (AlexNet, CaffeNet, VGG16) under data path fault injection and data buffer fault injection.
- Proposing a novel low-power fully-hardened SRAM cell for DNN accelerators to tolerate soft errors caused by SNUs and MNUs.
- Developing an integrated evaluation platform to assess the effectiveness of the proposed approach.

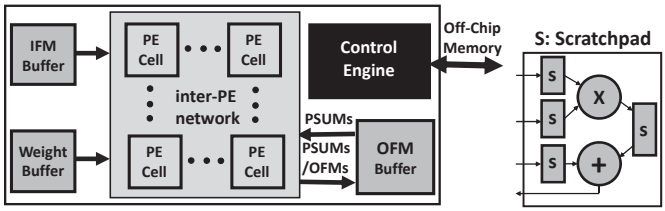
The rest of the paper is organized as follows. Section II provides the background on DNNs and DNN accelerators. Section III discusses the motivation for this work. In Section IV, we describe the proposed approach. Then, section V discusses evaluation methodology, implementation details, results and analysis. Finally, Section VI concludes this paper.

II. BACKGROUND

A. Deep Learning Neural Networks

Deep Learning neural networks (DNNs) are widely deployed in many fields [17]–[20] due to the achieved high accuracy. Among DNNs, convolutional neural networks (CNNs) show outstanding accuracy when performing tasks like classifications. A CNN is a directed acyclic graph by stacking multiple computational layers [21]. A higher abstraction of the input data, called feature maps (fmaps), is extracted to preserve fundamental and unique information.

Convolutional layers: The essential computation of a CNN mainly comes from convolutional (CONV) layers, which perform multi-dimensional convolutional operations. The modern CNNs can consist of three to hundreds of CONV layers [22]–[25]. In each CONV layer, multi-dimensional filters (also called kernels or weights) are applied on input feature maps (IFMs) to extract visual features to generate output feature maps (OFMs). Typically, generated results are being processed by an activation (ACT) function (e.g. ReLU) before becoming the input of the next layers.



(a) DNN Accelerator (b) PE cell

Fig. 1: DNN accelerator architecture.

Fully-Connected layers: A small number (e.g. 1-3) of fully-connected (FC) layers are usually stacked after the CONV layers to conduct the final classification tasks.

Pooling & Normalization layers: Between CONV layers and FC layers, optional layers, such as pooling (POOL), normalization (NORM), can be added. A POOL layer can reduce the size of feature maps by locally selecting the maximum input data and discarding the rest. The NORM layer averages its input data based on the surrounding input data.

The deployment procedure of CNNs is performed in two steps. First, the network is trained with a dataset for a specific application to achieve the expected accuracy. The training process is usually conducted off-line once as the huge amount of computation is very time-consuming. Second, the network is deployed into a platform performing inferences. The continuous flow of the input data is fed to the well-trained network to perform the real-time inference on each input data after deployment.

B. DNN Accelerator Architecture

In many deep neural network applications, inference operations should be performed within a fraction of second due to the maximum latency requirement (e.g. autonomous vehicles, IoT status monitoring) . However, DNNs are very computation and memory intensive [26], which makes general purpose platforms prone to be the bottleneck in processing DNNs [26]. Consequently, DNN accelerators are much needed to meet latency constraints in many applications. Current state-of-the-art DNN accelerators evaluate the network sequentially in a layer-by-layer fashion. A typical DNN accelerator has an array of processing elements (PEs) and various on-chip buffers. As shown in Fig. 1, the processing element (PE) arrays fetch input feature maps (IFMs) pixels (T_n pixels) from the input buffer, weights from the weight buffer, partial sums (PSUMs) from the output buffer, and then calculates PSUMs or output feature maps (OFMs) pixels (T_m pixels) that are stored in the output buffer.

Since the feature maps and weights of DNNs are very large, they cannot be fully stored on-chip. Consequently, they are held in the off-chip memory during each inference operation. However, due to the spatial and temporal localities which exist in feature maps and weights, data can be cached on-chip to be reused during each inference operation. Thus, large on-chip buffers are used to cache feature maps and weights on-chip in order to reduce the costly off-chip traffic between the accelerator and off-chip memory (e.g. Google's TPU has 28MB on-chip memory [1]). SRAMs are used to realize various on-chip buffers in DNN accelerators. Each PE cell

consists of a simple multiplier-and-adder and I/O scratchpads. The multiplier-adder performs multiplyaccumulate (MAC) operations during inference execution, and the I/O scratchpads store input pixels, weights and the partial sums during each MAC execution. The scratchpads are realized by small SRAM modules. PE cells communicate with each other through an inter-PE network.

III. Motivations

A. High Static Power in DNN Accelerators

As the downscaling technology has resulted in increasing leakage current in CMOS circuitry [8], [9], the impact on SRAM cells is especially remarkable due to the high transistor density in SRAM modules [8]. The leakage current can translate into temperature increase which further boosts the leakage current [8]. As explained above, DNN accelerators employ large SRAMs to realize on-chip buffers. In addition to on-chip buffers, a considerable part of the PE cell is dedicated to the scratchpads [2]. Therefore, by considering on-chip buffers and scratchpads, a significant part of the DNN accelerator is dedicated to the on-chip memory (e.g. near 75% of the Eyeriss DNN accelerator [2]). Besides, there is an increasing trend of using compact data representations in DNNs to improve the efficiency of computation and storage [11]. An example of this trend is binary neural networks (BNNs) [27]. Only two possible values (-1 or +1) are used for the pixels and weights in BBNs but they can still achieve an acceptable accuracy for many applications. In this kind of DNNs, the computational operations only need bit-wise XNORs for multiplication and a population counter for addition. This trend leads to memory-oriented DNN accelerators since the computational part of the accelerators is very simple and occupies less chip area [12]. Thus, SRAM modules are a predominant component in the current and future DNN accelerators, and contribute to a considerable amount of power consumption. The power consumption of the SRAM modules becomes more dominant when DNN accelerators employ compact data types.

B. Impact of Multiple-Node-Upset in DNN Accelerators

As modern Deep Learning Accelerators employ large on-chip buffers (i.e. SRAM modules) [1], [2] as well as some scratchpads inside the PE array, the accelerators are potentially vulnerable when soft errors occur in the SRAM cells. Traditionally, a major threat to the reliability of SRAM cells is the SNU caused by a particle strike. However, such a single particle strike has started to show an increasing probability of causing multiple-node-upsets in recent SRAM cells [3]–[5]. The reason behind this is that as the transistor dimensions and the distance between transistors shrink, the energy released by an incident ion remains constant. Thus, a cloud of charge from a particle may be shared across multiple nodes during a single event, potentially causing multi-node-upsets. This *charge sharing* phenomenon is more prominent in a SRAM module, as it has high density of transistors and its layout tends to use common or shared connections [4].

Depending on whether a multi-node upset occurs in multiple nodes of the same cell or in different cells, single-bit upsets (SBUs) or multi-bit upsets (MBUs) may occur in the on-chip memory of DNN accelerators, leading to soft errors. In our experiments of fault injection in DNN accelerators, we found that soft errors can lead to many mis-classification cases. These erroneous results may result in catastrophic consequences, especially in critical applications such as autonomous vehicles [6], unmanned aircrafts, and on-line status monitoring of nuclear power plants. Worse still, different from transient soft errors in many other systems, soft errors in DNN accelerator buffers may have a more persistent effect as they can be reused many times.

C. Limitation of Existing Reliability Techniques

There is a large body of research dedicated to improving the reliability of SRAMs. However, in general, the reliability techniques which can be applied to DNN accelerators can be categorized into three main topics: Error Correction Code (ECC) techniques, spatial redundancy techniques, and hardened SRAM cells.

ECC Techniques in DNN Accelerators: ECC has a limited number of MBU detection capability, while the probability of MBU increases as CMOS technologies move to smaller sizes for a denser integration [28]. Thus, this technique cannot provide highly reliable infrastructure for SRAMs in nano-scaled CMOS technologies. Furthermore, ECC techniques have performance penalty since additional circuits should be added to perform bit-checking and correction during write and read operations of the buffers and the scratchpads [29].

Spatial Redundancy Techniques: In this class of techniques, DNN accelerators are implemented using spatial redundant methods, such as triple modular redundancy (TMR) or N-MR (N-modular redundancy), to tolerate soft errors. The spatial redundancy techniques can be applied to DNN accelerators at the software level or hardware level [6]. Although spatial redundancy techniques can increase the reliability of DNN accelerators, these techniques are very expensive in terms of area, power and speed due to the extra resources required for implementing the redundancy. Another main drawback of the spatial redundancy of techniques is the limited capability of tolerating MBU, which may not be sufficient for contemporary and future DNN accelerators that are manufactured in nano-scaled technologies. For example, TMR can mask only one fault, and the majority voter may fail in determining the correct output in the presence of MBU [30]. Additionally, spatial redundancy techniques usually have slower timing due to the delay of the voter at the output stage [31].

Fully-Hardened SRAM Cells: Because of the abovementioned drawbacks, fully-hardened SRAM cells become more attractive to be used as the memory cells in DNN accelerators compared with other soft error mitigation techniques. In general, using hardened SRAM cells to tolerate bit upsets is an approach with less area, power, and delay [13]. Fully-hardened SRAM cells can tolerate SNU when either one or zero is stored in the memory cell. They provide full protection against SNU. However, none of the existing fully-hardened SRAM cells can tolerate MNU which is a serious challenge in nano-scaled CMOS technologies. Furthermore, although fully-hardened SRAM cells provides a power efficient approach to address the soft errors, most of the existing fully-hardened SRAM cells still consume considerable leakage current (more details in the evaluation section).

D. Zero-Biased Data in DNNs

Prior studies show that there is a strong bias towards zero both in feature maps [2] [14] and weights [15] across different networks. For example, about 40% of feature maps in the second CNN layer of the AlexNet are zeros [2]. This number increases rapidly as the accelerator moves to deeper layers: for instance, on average 75% of the feature maps are zeros in the fifth CNN layer of AlexNet [2]. This high percentage of zeros in feature maps is resulted from the adoption of ReLU function: this function filters out all the pixels with a negative value in feature maps to zeros. Similar behavior is observed for weights in prior work. That is, the majority of the weights that are proved to be not important in the CNN layers can be filtered out to zeros without losing accuracy [15]. For instance, a recent work reports that for some layers of AlexNet and VGGNet-D, the sparsity can reach up to 91% and 96%, respectively [15].

In addition to the high percentage of zeros in feature maps and weights, a recent study [6] as well as our experiments show that bit-upsets from zero to one are more likely to lead to false outputs in comparison with the bit-upsets from ones to zeros. Therefore, there is an opportunity that takes advantage of the above two bias types in order to reduce the static power consumption and soft error rate of the on-chip memory in DNN accelerators.

IV. Proposed Approach

In this work, we focus on designing an SRAM cell customized for DNN accelerators, as fully-hardened SRAM cells in DNN accelerators are more attractive than other reliability techniques due to the lower overhead in area, power and delay. The proposed SRAM cell is designed based on the observation that there is a strong bias towards zero in feature maps and weights. A novel Zero-Biased MNU-Aware SRAM cell (ZBMA) architecture is proposed, which can tolerate MNU for zeros, and SNU for both zeros and ones. In the meantime, the proposed cell has a reduced leakage current compared with the conventional (CV) SRAM cell and state-of-the-art fully-hardened SRAM cells. The proposed SRAM cell can be used both in buffers and scratchpads of DNN accelerators.

Fig. 2 shows an overview of using the proposed SRAM cell in the on-chip memories of DNN accelerators. As shown in Fig. 2a for the OFM buffer and the scratchpads of a PE cell, the SRAMs in DNN accelerators should be partitioned into smaller memory banks (represented as horizontal dark boxes in the Fig. 2a) in order to avoid long word-lines and bit-lines [32]. Long word-lines and bit-lines have high capacitance which causes long delay and high energy consumption [32]. This is especially important in DNN accelerators as during network processing a large number of on-chip memory accesses are generated by the PE cells [2]. Fig. 2b show the architecture of an SRAM bank. Generally, each SRAM bank consists of an array of memory cells, row & column decoders, and column & bit-line pre-charge circuitries. Each cell in the array stores one bit, and it is realized by the proposed SRAM cell as shown in Fig. 2c. The *N*-bit address is decoded by the row and column decoders to select *m* cells during read and write operations. The pre-charge circuitries are used to pre-charge bit-lines during read operations. The column circuitries contain amplifiers and write-buffers to sense or send the data from or to selected cells during read and write operations (more details on read and write operations are provided later in this section).

A. Novel Memory Cell for DNN Accelerators

To address the MNU and high static power issues in DNN accelerators, our objective is to develop a low-power SRAM cell that allows SNUs and MNUs to be tolerated directly within the memory array (Fig. 2b) instead of relying on higher levels of abstraction. The challenge of achieving this SRAM cell is in the positive feedback between the nodes in a SRAM cell when there are upsets at more than one node. This positive feedback is used to store the data in the SRAM cell. However, it is the main reason of bit upset at MNU-occurrences in existing SRAM cells, even if they are fully hardened against SNUs. The Zero-Biased MNU-Aware SRAM cell is shown in Fig. 2c. The proposed SRAM cell consists of two storage cores including an upper core and a lower core (the cores are labeled in Fig. 2c). This redundancy in cell circuit is needed to recover any corrupted nodes by the unaffected nodes. Similar to a CV SRAM cell, there are two complementary storage nodes in each core, and the stored bit can be written into the cell by the access transistors (M20 and M21 in Fig. 2c). In order to tolerate SNU (for zeros and ones) and MNU (for zeros), each core is designed based on the following rationales.

First, to eliminate the propagation of node upsets from an affected node to its complement in the same core, nodes in one core are driving the driver transistors of nodes in the opposite core. For example, nodes ST and STB drive the driver transistors (M10 and M11) of node A. In this way, a node upset (e.g., at node A) will not affect its complement (node B).

Second, when a particle hits the drain/source of an 'OFF' transistor, it creates the unwanted SNU or MNU at a node or nodes of the circuit. The polarity of the SNU and the MNU (positive or negative) depends on the type of the transistor (PMOS or NMOS). Thus, two cases are possible: (1) if a particle strikes an 'OFF' PMOS transistor, a positive ($0 \rightarrow 1$) SNU or MNU is injected to a node or nodes; (2) if a particle strikes an 'OFF' NMOS transistor, a negative SNU or MNU ($1 \rightarrow 0$) is injected to a node or nodes. In each core of the proposed memory cell, we use NMOS transistors to drive the storage nodes (i.e., A and B in the lower core, and ST and STB in the upper core). With this design, positive upsets cannot happen at any node of cores. For example, when zero is stored in the cell, the only possible MNU is from simultaneous negative upsets at both STB and B. In addition, since NMOS transistors are used to drive the storage nodes, a negative upset at a node cannot turn ON any NMOS driver transistors at the storage nodes to change the voltage of the nodes. For instance, when zero is stored in the cell, a negative upset at B cannot turn ON any NMOS driver transistors at the storage nodes. It only turns OFF transistors M3 and M6, and nodes ST and STB will be floated temporarily. Thus, the voltages of nodes ST and STB stay steady.

Third, besides using NMOS transistors at sensitive nodes to tolerate MNU when zero is stored in the cell, node STB is used to drive the pull-up network of node ST in the unidirectional way using an inverter (M17, M18, and M19). Furthermore, the output of the inverter that drives the gate of transistor (M2) in

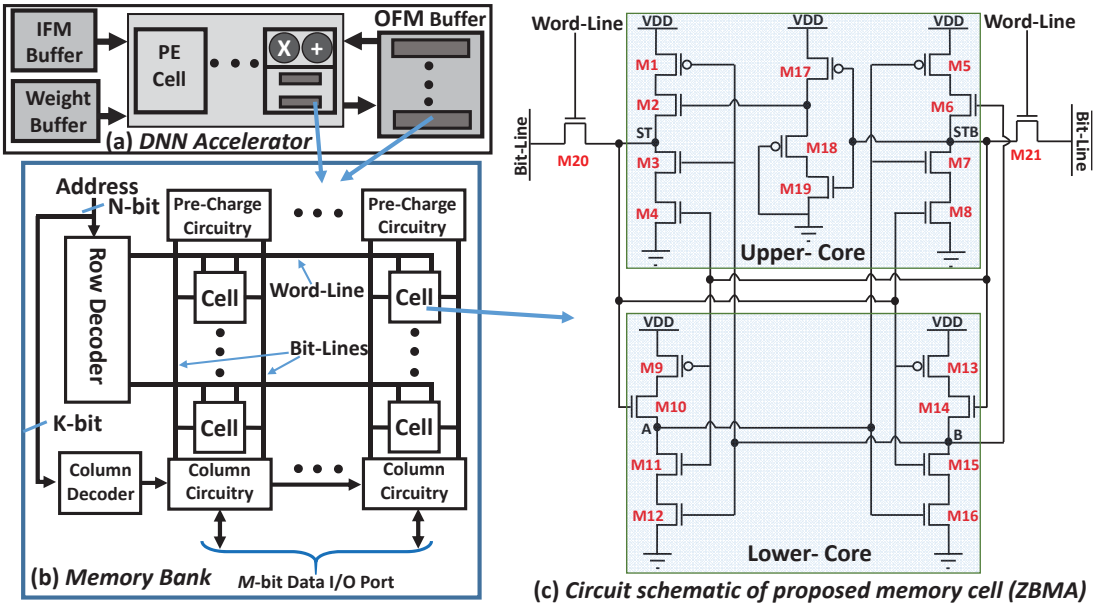


Fig. 2: On-chip memories in DNN accelerators based on the proposed memory cell.

the pull-up network is driven by the PMOS transistors. In other words, the output of the inverter that generates the control signal of pull-up network is '1' hardened (i.e., negative upset cannot happen at output). The resulting SRAM cell is a fully-hardened SRAM cell which can tolerate MNU when zero is stored in the cell.

B. Node Upsets in Memory Cell

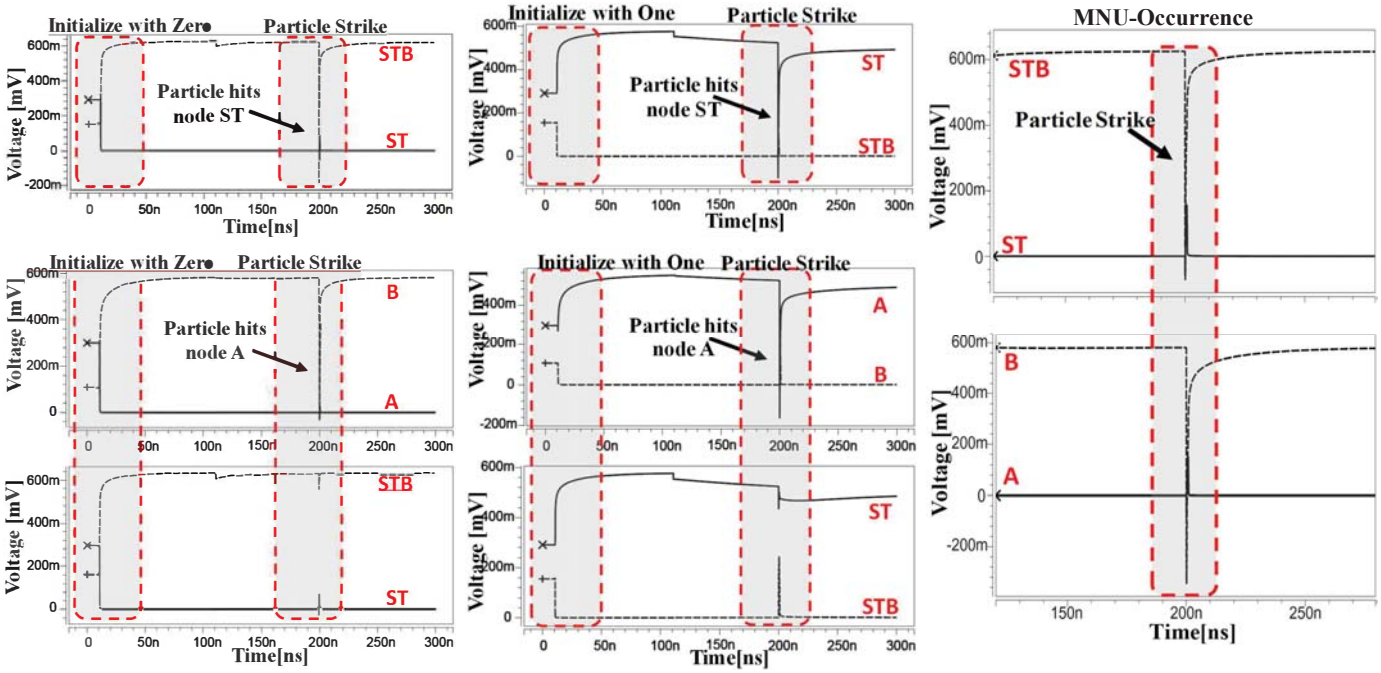
When a particle hits a cell in the memory array, it creates an unwanted SNU at a node in the struck memory cell. However, as discussed in the motivation section, a particle hit may lead to MNU which affects multiple nodes of the cell due to charge sharing. Generally, four possible cases can occur when a particle strikes a node in the proposed cell as following: **1)-**Zero is stored in the cell and a negative SNU is injected at the node STB *or* the node B. **2)-**One is stored in the cell and a negative SNU is injected at the node ST *or* the node A. **3)**-Zero is stored in the cell and a MNU is injected at the node STB *and* the node B. **4)-**One is stored in the cell and a MNU is injected at the node ST *and* A. Fig. 3 shows the HSPICE simulated waveforms for case 1, 2 and 3. The simulations are performed using 16nm PTM High-Performance CMOS technology [33] at the supply voltage of 0.7V. As can be seen from the simulated waveforms, the proposed SRAM cell can eliminate SNU when zero or one is stored in the cell. Also, it can eliminate the MNU when zero is store in the cell. However, when one is stored in the cell and a MNU occurs, the cell flips to zero.

C. Static Power Reduction

In order to maintain the performance of transistors when employing the low-supply voltage in downscaled technologies, the threshold voltage (V_T) of transistors should be scaled down correspondingly. However, the sub-threshold leakage current exponentially increases as the threshold voltage decreases. Consequently, a large amount of power (static power) is dissipated due to sub-threshold leakage current. In the proposed SRAM cell, high-V_T PMOS transistors (M1, M5, M9, and M13 in Fig. 2c, called as cut-off transistors) are inserted between VDD and pull-up networks to cut off the VDD when the nodes connected to the pull-up networks are in zero state. Similarly, high-V_T NMOS transistors (M4, M8, M12, and M16 in Fig. 2c) are inserted between GND and pull-down networks to cut off the GND when nodes connected to pull-down networks are in one state. In addition to these high-V_T cut-off transistors, wires between nodes and gates of the transistor are smartly connected in order to turn off the transistor in the pull-up network when the pull-down network is conducting and vise versa. For example, when zero is stored at node A, VDD is disconnected by the cut-off PMOS transistor M9, and the M10 located in the pull-up network of node A is also turned off. In this way, a stack of 'OFF' transistors are created between the power supply and the nodes of the cell. A stack of 'OFF' transistors reduce the sub-threshold leakage current significantly [34]. Fig. 4 shows all stacks of 'OFF' transistors in the core of the proposed SRAM cell when zero is stored (highlighted by green color). The combination of stacks of 'OFF' transistors and high-V_T cut-off transistors greatly reduce the leakage current in the proposed SRAM cell (as shown later in the evaluation).

D. Transistor Sizing and Scalability of Memory Cell

The flow of read and write operations in the proposed SRAM cell is similar to the CV SRAM cell. That is, for a write operation, the data and its compliment are driven (by the column circuitry in Fig. 2b) on the bit-line and bit-line-bar, respectively. Then, the word-line is asserted to VDD in order to write the data into the cell. For a read operation, both the bit-line and bit-line-bar are pre-charged to VDD (by the pre-charge circuitry in Fig. 2b). Then, the word-line is asserted to VDD in order to connect the bit-lines to the cell. Right before the word-line assertion, the bit-lines



(a) Case 1: SNU on node STB or node B (b) Case 2: SNU on node ST or node A (c) Case 3: MNU due to charge sharing on node STB or node B

Fig. 3: Node upsets in the proposed memory cell.

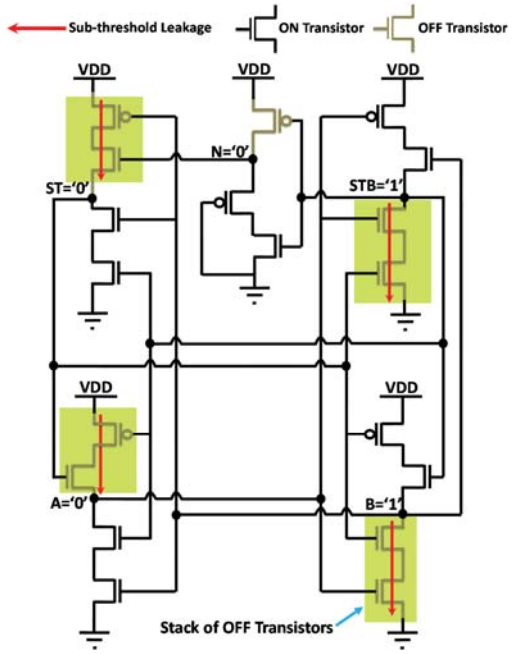


Fig. 4: Stacks of OFF transistors when zero is stored.

are left floated. Thus, depending on the stored value in the cell, one of the bit-lines is discharged to a lower voltage. Then, the voltage difference between bit-lines is sensed by a differential sense amplifier which is located inside the column circuitry. To achieve correct read and write operations in the CV SRAM cell, a constraint on transistor sizes should be followed [32]. However, the correct read and write operations in the proposed SRAM cell can be achieved using minimum transistor sizes. In the meantime, the immunity of the proposed SRAM cell against SNUs and MNUs does not depend on the size of the transistors. Therefore, for scaling from a given technology to a smaller technology, the minimum possible size can be considered for the transistors in the cell. The minimum possible size is used for transistors among all the particle strike simulations in Fig. 3. The results show that the proposed cell can provide the full protection for SNUs (both zero and one) and MNUs (for zero) with the correct read and write operations when the minimum size is used for all the transistors.

V. Evaluation

We evaluate the effectiveness of the proposed approach using a combination of CAD tools as well as in-house developed tools. Fig. 5, shows the connection between different tools used in this work. First, the layouts of memory cells are designed in Cadence Virtuoso. Then, based on the layouts, the required netlists are extracted. Significant effort is spent on developing a tool that can inject single or multiple simultaneous upsets into one or multiple nodes in the netlist, so as to mimic particle strikes and charge-sharing. The power, delay, and immunity against SNUs and MNUs are measured through HSPICE simulations.

An optimization program is developed to generate optimized parameters (T_n and T_m) for the accelerators. After inputing CNN layer descriptions, chip die area requirement, target frequency, data type format, maximum memory bandwidth, and memory cell characteristics (obtained from the Cadence

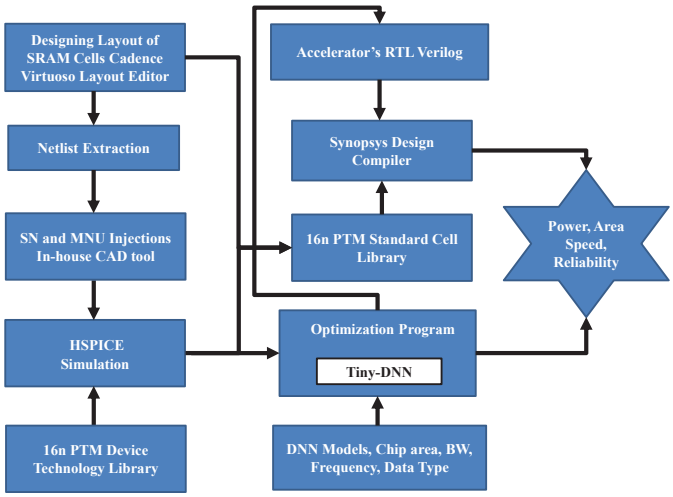


Fig. 5: Simulation environment for evaluation.

TABLE I: Evaluated DNNs

Network	Topology	Dataset	label No.	Data Type
CaffeNet	5 CONV + 3 FC	ImageNet	1000	FLOAT32
AlexNet	5 CONV + 3 FC	ImageNet	1000	FLOAT32
VGG16	13 CONV + 3 FC	ImageNet	1000	FLOAT32

Virtuoso and the HSPICE simulations), the program automatically calculates the expected execution cycles, memory bandwidth, number of PE cells and required on-chip memory. For DNN reliability evaluation, the tool maps each line in the Tiny-DNN [16], a DNN framework, to the corresponding hardware components of DNN accelerators. The Tiny-DNN is modified to enable the fault injections in different components of DNN accelerators.

The parametrized DNN accelerator is implemented in RTL Verilog, whose architecture is based on the paper [35]. Parameters are obtained from the optimization program based on different network configurations. Synopsys Design Compiler is used to synthesize the RTL implementation with 16nm PTM cell library [36]. The library is updated to reflect the characteristics of different memory cells.

A. Error Propagation Analysis of DNN Accelerators

1) Experiment Setup: We use Silent Data Corruption (SDC) probability to evaluate the reliability of DNN accelerators. The SDC probability is defined as the probability of SDC that affect the visible state of the program under given soft errors. The definition is in line with the one in other papers [6], [37]. A SDC is considered to happen if the top ranked element that is predicted by the accelerator with soft errors is different from the one that is predicted in the fault-free accelerator, as the first predicted element is usually used for the subsequent operations in the systems. The networks with different topologies but covering common features of modern DNNs should be tested. The Table I lists DNNs evaluated in this paper. The parameters of these networks are publicly available online.

Consistent with the paper [6], we consider two types of fault injection: data path (i.e. scratchpads inside PE cells) and

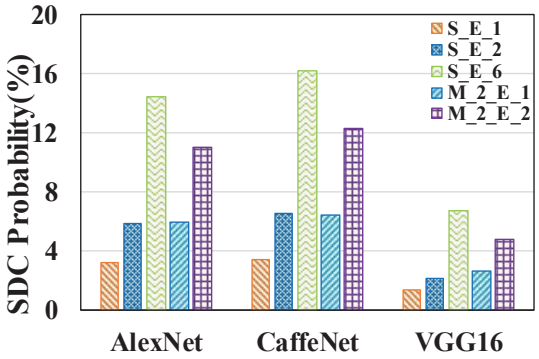


Fig. 6: The SDC probability of data path soft error injection.

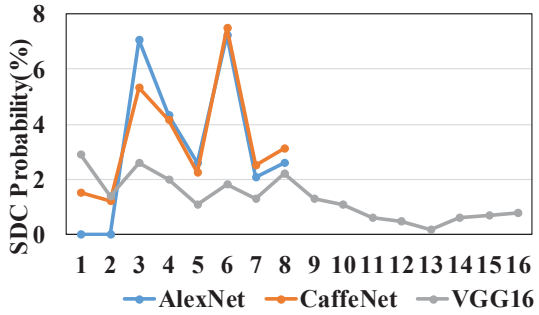


Fig. 7: The SDC probability of each layer (S_E_1).

buffers. Combinational logics and control logics are not considered as they are not sensitive to soft errors compared with storage units [38]. As analyzed above, the particle strike can result in multiple-bit-upsets in current downscaled technology, so we test multiple error situations: single-node upset, two-node upsets, and six-node upsets are evaluated for a given layer; and for two layers, only single-node upsets and two-node upsets per layer are tested, as it is very unlikely to have six errors across different layers in one inference processing. Each situation of fault injection is run for 3000 times, and the fault injection is in accordance with related research [6], [37].

2) Fault Injection in Datapath: Since scratchpads in PE cells are frequently read and written, the occurrences of soft errors have only transient impact on the calculation process. Fig. 6 shows the results of different injection types in the accelerator data path. S represents single layer, while M stands for multiple layers. E is the numbers of error injection. For instance, M_2_E_2 means that two layers are selected to have fault injections, and each layer is injected with two errors. In general, the increase of soft error injection leads to higher SDC probability among all tested DNNs. The SDC probability can reach 16.20% when six errors occur in only one layer. The most likely explanation is that the increased number of errors are more likely to cause large deviations of the key features, which further affects the final classification. From the figure, we can also observe that as the network going deeper, the impact brought by soft errors is mitigated, as deeper DNNs are more immune to the errors [6]. However, the adopted DNNs in IoT or other real-time recognition devices are often

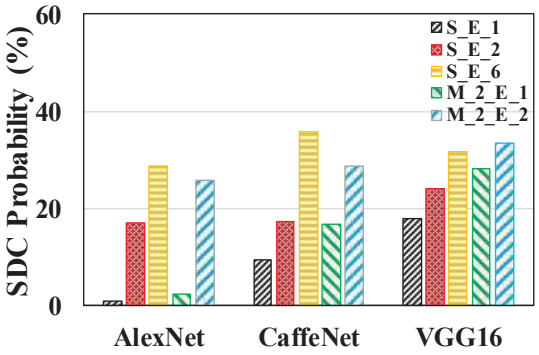


Fig. 8: The SDC probability of soft error injection in data buffers.

TABLE II: Comparison of fully-hardened SRAM cells

SRAM Cells	Refresh Signal	Full protection against SNU	Zero Full protection against MNU	Year
DICE [39]	No	Yes	No	1996
RSC-1 [40]	Yes	Yes	No	2011
RSC-2 [41]	Yes	Yes	No	2012
RSC-3 [42]	Yes	Yes	No	2014
RHM-1 [13]	No	Yes	No	2014
RHM-2 [43]	No	Yes	No	2015
RHD-1 [44]	Yes	Yes	No	2015
RHD-2 [45]	No	Yes	No	2016
RHBD [46]	No	Yes	No	2017
ZBMA	No	Yes	Yes	2018

shallow due to computation limitation and power constraints. Hence, developing more reliable accelerators especially for online safety-critical applications is crucial. Fig. 7 plots the SDC probability of each layer under the S_E_1 condition. Since AlexNet and CaffeNet employ NORM Layers after their first two CONV layers separately, the SDC probabilities of first two layers are much lower than other layers. This is because the NORM layer normalizes the output of the first and second layer, which mitigates the effect of large deviations in outputs. Due to the lack of the NORM layers in VGG16, the SDC probability of each layer is close to each other, and its line seems flatter than other two networks.

3) Fault Injection in Data Buffers: Modern deep learning accelerators employs large on-chip buffers to reduce off-chip memory accesses. The IFMs and weights are partially loaded into the buffers and reused for several times. Fig. 8 illustrates the SDC probability of the soft injection in data buffers. The IFMs, weights, OFMs are randomly selected to be injected with errors. We can observe that the SDC probability is much higher than the one under data path injection, as the soft errors occurring in the buffer are reused for several times, and can spread to different locations within a short time, leading to more opportunities to generate the false output. Furthermore, the deeper network in this test does not show higher ability of error tolerance, since the buffer-reuse technique also propagates the errors to multiple channels, which is likely to cause large deviation to the key features. Therefore, though buffer-reuse greatly reduces off-chip traffic, it also increases the vulnerability of DNN accelerators to soft errors.

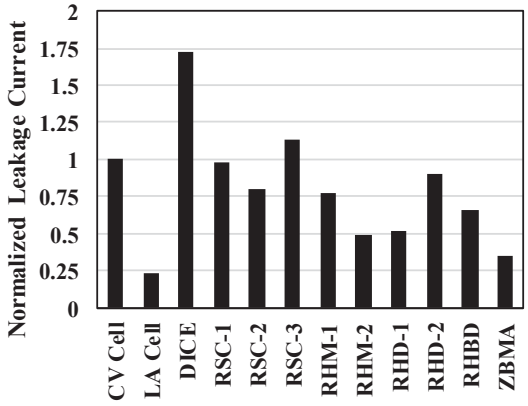


Fig. 9: Normalized leakage current of different fully-hardened SRAM cells (to the CV SRAM cell).

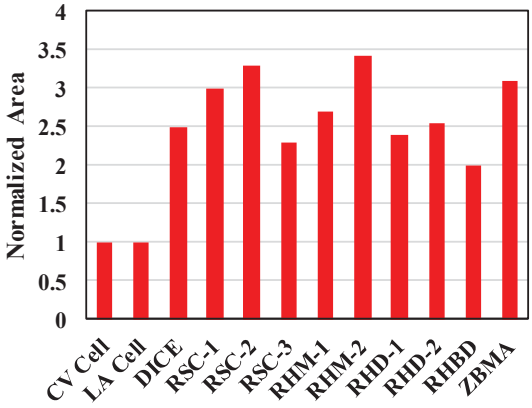
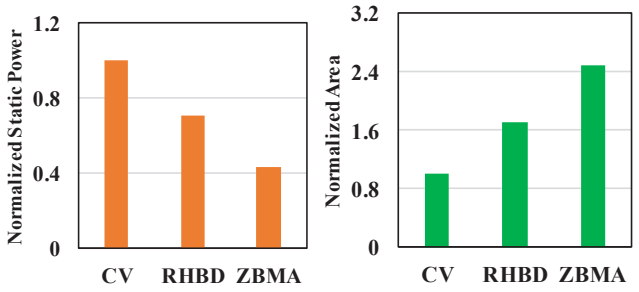


Fig. 10: Normalized area of different fully-hardened SRAM cells (to the CV SRAM cell).

B. Memory Cell Characteristics in DNN Accelerators

In the past two decades, many partially and fully-hardened SRAM cells are proposed which can be used as the memory cell in the DNN accelerators. Partially hardened SRAM cells such as [47], [48] can only tolerate SNUs for single logical value (zero or one). However, fully-hardened SRAM cells provide full protection against SNUs. Therefore, for reliable applications, fully-hardened SRAM cells are more attractive than other types of hardened SRAM cells because of their full immunity against SNUs. Table II compares different well-known fully-hardened SRAM cells which have been proposed in the past two decades. In some fully-hardened SRAM cells [40]–[42], loop cutting technique is used to cut off the positive feedback loop in the holding mode. In this way, a transient glitch (upset) cannot be propagated along feedback loop to its starting point in order to change the content of the cell [40]. This type of fully-hardened SRAM cells use a refresh signal to provide guarantee for maintaining the correct data [49]. Refresh signal must be applied and routed to all memory cells in the accelerator. Thus, some extra peripheral circuitry is also needed to generate the refresh signal [49]. Each refresh cycle includes charging and discharging the load capacitance of the refresh line. As the refresh line is routed for all memory



(a) Normalized static power (b) Normalized Area

Fig. 11: Static power and area of the DNN accelerators based of the different SRAM cells (Normalized to the DNN accelerator based on the CV SRAM cell).

cells, the load capacitance on the refresh line is typically very large, and a significant amount of power may be dissipated in the refresh cycle [34]. As exhibited in Table II, none of the existing fully-hardened SRAM cells can tolerate MNU when zero is stored in the cell. However, the proposed SRAM cell (ZBMA) can provide full protection against SNUs and tolerate MNUs for logic zero without using any refresh signal.

Fig. 9 shows the normalized average leakage current of different fully-hardened SRAM cells as well as the proposed SRAM cell. As it is shown in Fig. 9, most of existing fully-hardened SRAM cells consume quite amount of leakage current in comparison with the proposed SRAM cell. The proposed SRAM cell has the lowest leakage current among all fully-hardened SRAM cells. In addition to leakage current of existing fully-hardened SRAM cells, Fig. 9 shows the average leakage current for a low-leakage asymmetric SRAM cell (LA cell) [50]. This SRAM cell is an optimized version of CV SRAM cell which reduces the leakage current for zero logic using high-V_T transistors. While this cell has the lowest leakage current, it does not provide any protection against the SNUs and the MNUs. The proposed SRAM cell is able to reduce leakage current by 2.84X over the CV SRAM cell, 4.92X over DICE [39], 2.8X over RSC-1 [40], 2.26X over RSC-2 [41], 3.23X over RSC-3 [42], 2.19X over RHM-1 [13], 1.39X over RHM-2 [43], 1.47X over RHD-1 [44], 2.58X over RHD-2 [45], and 1.88X over RHBD [46]. The leakage currents in Fig. 9 are obtained from HSPICE simulations using the minimum possible transistor sizes.

Fig. 10 compares the area overhead of different fully-hardened SRAM cells and the proposed SRAM cell under the same design rules. All the area are normalized to the CV SRAM cell. The area overhead in the Fig. 10 is obtained from Cadence Virtuoso as well as other related work [42], [44]–[46].

C. DNN Accelerators Comparison

We also examine the impact of different SRAM cells (CV cell, state-of-the-art fully-hardened cell, and proposed cell) on different metrics of DNN accelerators by targeting the acceleration of AlexNet in 32-bit floating-point at 150MHz. The optimized parameters for accelerators are generated by the optimization program tool mentioned in earlier. The accelerators have three different buffers including the weights buffer, the IFMs buffer, and the OFMs buffer. All the buffers in the accelerators use standard double buffering techniques to hide off-chip communication latency with computation. The generated parameters are used by the RTL Verilog implementation of each accelerator. The RTL Verilog implementations are synthesized under Synopsys Design Compiler to report different metrics. For a fair comparison, all the accelerators are optimized to have a same performance (141.66 GOPS) and off-chip traffic (1.87 GB/s). Fig. 11 shows the normalized static power and area of DNN accelerators based on different SRAM cells. As can be seen, using the proposed SRAM cell (ZBMA), the static power of DNN accelerator can be reduced by 2.6X and 1.79X compared with the CV cell and the state-of-the-art hardened cell (RHBD), respectively.

The accelerator based on RHBD cannot tolerate MNUs while the accelerator based on ZNMA can tolerate MNUs when zero is stored. Using RHBD, we observe that the SDC of MNUs due to in-buffer MNUs of the AlexNet accelerator can reach up to 28.88%. However, when ZBMA is used, this percentage is reduced to 0.01%, which is a huge improvement in the immunity of the AlexNet accelerator against the MBUs.

VI. Conclusion

DNNs have been successfully deployed in many fields and domains, including safety-critical applications such as autonomous vehicles and unmanned aircrafts. In this paper, we analyze the impact of soft errors on the reliability of DNN accelerators through three popular DNNs, and propose a novel SRAM cell design to eliminate soft errors while achieving a low power consumption. The proposed SRAM cell takes into consideration the bias towards zero in feature maps and weights of DNNs to reduce the leakage current and achieve a highly reliable inference process. A simulation platform is developed to study the impact of memory cells on the reliability and power consumption of the DNN accelerators. Simulation results show that the DNN accelerator based on the proposed approach offers significant advantage in reliability and power-efficiency, compared with other SRAM cell designs.

VII. Acknowledgement

We sincerely thank the reviewers for their helpful comments and suggestions. This research was supported, in part, by the National Science Foundation (NSF) grants #1566637, #1619456 and #1750047.

References

[1] N. P. Jouppi, C. Young, N. Patil, D. Patterson, G. Agrawal, R. Bajwa, S. Bates, S. Bhatia, N. Boden, A. Borchers *et al.*, "In-datacenter performance analysis of a tensor processing unit," in *International Symposium on Computer Architecture (ISCA'17)*. ACM, 2017.

[2] Y.-H. Chen, T. Krishna, J. S. Emer, and V. Sze, "Eyeriss: An energy-efficient reconfigurable accelerator for deep convolutional neural networks," *IEEE Journal of Solid-State Circuits*, 2017.

[3] N. M. Atkinson, J. R. Ahlbin, A. F. Witulski, N. J. Gaspard, W. T. Holman, B. L. Bhuva, E. X. Zhang, L. Chen, and L. W. Massengill, "Effect of transistor density and charge sharing on single-event transients in 90-nm bulk cmos," *IEEE Transactions on Nuclear Science*, 2011.

[4] J. D. Black, P. E. Dodd, and K. M. Warren, "Physics of multiple-node charge collection and impacts on single-event characterization and soft error rate prediction," *IEEE Transactions on Nuclear Science*, 2013.

[5] T. Loveless, S. Jagannathan, T. Reece, J. Chetia, B. Bhuva, M. McCurdy, L. Massengill, S.-J. Wen, R. Wong, and D. Rennie, "Neutron-and proton-induced single event upsets for d-and dice-flip/flop designs at a 40 nm technology node," *IEEE Transactions on Nuclear Science*, 2011.

[6] G. Li, S. K. S. Hari, M. Sullivan, T. Tsai, K. Pattabiraman, J. Emer, and S. W. Keckler, "Understanding error propagation in deep learning neural network (dnn) accelerators and applications," in *International Conference for High Performance Computing, Networking, Storage and Analysis*. ACM, 2017.

[7] N. D. Lane, S. Bhattacharya, P. Georgiev, C. Forlivesi, and F. Kawsar, "An early resource characterization of deep learning on wearables, smartphones and internet-of-things devices," in *International Workshop on Internet of Things towards Applications*. ACM, 2015.

[8] A. Calimera, A. Macii, E. Macii, and M. Poncino, "Design techniques and architectures for low-leakage srams," *IEEE Transactions on Circuits and Systems I: Regular Papers*, 2012.

[9] A. Islam and M. Hasan, "Leakage characterization of 10t sram cell," *IEEE transactions on electron devices*, 2012.

[10] A. Agarwal, C. H. Kim, S. Mukhopadhyay, and K. Roy, "Leakage in nano-scale technologies: mechanisms, impact and design considerations," in *Proceedings of Design Automation Conference (DAC'04)*. ACM, 2004.

[11] E. Nurvitadhi, G. Venkatesh, J. Sim, D. Marr, R. Huang, J. Ong Gee Hock, Y. T. Liew, K. Srivatsan, D. Moss, S. Subhaschandra *et al.*, "Can fpgas beat gpus in accelerating next-generation deep neural networks?" in *International Symposium on Field-Programmable Gate Arrays (FPGA'17)*. ACM, 2017.

[12] S. Takamaeda-Yamazaki, K. Ueyoshi, K. Ando, R. Uematsu, K. Hirose, M. Ikebe, T. Asai, and M. Motomura, "Accelerating deep learning by binarized hardware," in *Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC'17)*, 2017.

[13] J. Guo, L. Xiao, and Z. Mao, "Novel low-power and highly reliable radiation hardened memory cell for 65 nm cmos technology," *IEEE Transactions on Circuits and Systems I: Regular Papers*, 2014.

[14] J. Albericio, P. Judd, T. Hetherington, T. Aamodt, N. E. Jerger, and A. Moshovos, "Cnvlutin: Ineffectual-neuron-free deep neural network computing," in *ACM SIGARCH Computer Architecture News*, 2016.

[15] S. Han, H. Mao, and W. J. Dally, "Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding," *arXiv preprint arXiv:1510.00149*, 2015.

[16] "Tiny-dnn," https://github.com/tiny-dnn/tiny-dnn, 2016.

[17] C. Chen, A. Seff, A. Kornhauser, and J. Xiao, "Deepdriving: Learning affordance for direct perception in autonomous driving," in *International Conference on Computer Vision (ICCV'15)*. IEEE, 2015.

[18] M. Song, K. Zhong, J. Zhang, Y. Hu, D. Liu, W. Zhang, J. Wang, and T. Li, "In-situ ai: Towards autonomous and incremental deep learning for iot systems," in *International Symposium on High Performance Computer Architecture (HPCA'18)*. IEEE, 2018.

[19] B. Pourbabaee, M. J. Roshtkhari, and K. Khorasani, "Deep convolutional neural networks and learning ecg features for screening paroxysmal atrial fibrillation patients," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 2017.

[20] M. G. Fernandez, A. Tokuhiro, K. Welter, and Q. Wu, "Nuclear energy systems behavior and decision making using machine learning," *Nuclear Engineering and Design*, 2017.

[21] Y. LeCun, K. Kavukcuoglu, and C. Farabet, "Convolutional networks and applications in vision."

[22] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012.

[23] L. Gatys, A. S. Ecker, and M. Bethge, "Texture synthesis using convolutional neural networks," in *Advances in Neural Information Processing Systems*, 2015.

[24] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, A. Rabinovich *et al.*, "Going deeper with convolutions." CVPR, 2015.

[25] K. He, X. Zhang, S. Ren, and J. Sun, "Identity mappings in deep residual networks," in *European Conference on Computer Vision*, 2016.

[26] Y. Shen, M. Ferdman, and P. Milder, "Maximizing cnn accelerator efficiency through resource partitioning," in *International Symposium on Computer Architecture (ISCA'17)*. ACM, 2017.

[27] M. Rastegari, V. Ordonez, J. Redmon, and A. Farhadi, "Xnor-net: Imagenet classification using binary convolutional neural networks," in *European Conference on Computer Vision*. Springer, 2016.

[28] S. M. Jahinuzzaman, D. J. Rennie, and M. Sachdev, "A soft error tolerant 10t sram bit-cell with differential read capability," *IEEE Transactions on Nuclear Science*, 2009.

[29] M. Nicolaidis, "Design for soft error mitigation," *IEEE Transactions on Device and Materials Reliability*, 2005.

[30] J. Tarrillo, F. L. Kastensmidt, P. Rech, C. Frost, and C. Valderrama, "Neutron cross-section of n-modular redundancy technique in sram-based fpgas," *IEEE Transactions on Nuclear Science*, 2014.

[31] K. S. Morgan, D. L. McMurtrey, B. H. Pratt, and M. J. Wirthlin, "A comparison of tmr with alternative fault-tolerant design techniques for fpgas," *IEEE transactions on nuclear science*, 2007.

[32] N. H. Weste and y. Harris, David', *CMOS VLSI design: a circuits and systems perspective*.

[33] Predictive Technology Model, http://ptm.asu.edu/.

[34] A. A. Mazreah and M. T. M. Shalmani, "Low-leakage soft error tolerant port-less configuration memory cells for fpgas," *INTEGRATION, the VLSI journal*, 2013.

[35] C. Zhang, P. Li, G. Sun, Y. Guan, B. Xiao, and J. Cong, "Optimizing fpga-based accelerator design for deep convolutional neural networks," in *Proceedings of the 2015 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays*. ACM, 2015.

[36] http://sportlab.usc.edu/downloads/packages/.

[37] J. Wei, A. Thomas, G. Li, and K. Pattabiraman, "Quantifying the accuracy of high-level fault injection techniques for hardware faults," in *International Conference on Dependable Systems and Networks (DSN'14)*. IEEE, 2014.

[38] B. Gill, N. Seifert, and V. Zia, "Comparison of alpha-particle and neutron-induced combinational and sequential logic error rates at the 32nm technology node," in *Reliability Physics Symposium, 2009 IEEE International*. IEEE, 2009.

[39] T. Calin, M. Nicolaidis, and R. Velazco, "Upset hardened memory design for submicron cmos technology," *IEEE Transactions on Nuclear Science*, 1996.

[40] S. Lin, Y.-B. Kim, and F. Lombardi, "A 11-transistor nanoscale cmos memory cell for hardening to soft errors," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 2011.

[41] S. Lin, Y. B. Kim, and F. Lombardi, "Analysis and design of nanoscale cmos storage elements for single-event hardening with multiple-node upset," *IEEE Transactions on Device and Materials Reliability*, 2012.

[42] H.-B. Wang, J.-S. Bi, M.-L. Li, L. Chen, R. Liu, Y.-Q. Li, A.-L. He, and G. Guo, "An area efficient seu-tolerant latch design," *IEEE Transactions on Nuclear Science*, 2014.

[43] J. Guo, L. Xiao, T. Wang, S. Liu, X. Wang, and Z. Mao, "Soft error hardened memory design for nanoscale complementary metal oxide semiconductor technology," *IEEE Transactions on Reliability*, 2015.

[44] R. Rajaei, B. Asgari, M. Tabandeh, and M. Fazeli, "Design of robust sram cells against single-event multiple effects for nanometer technologies," *IEEE Transactions on Device and Materials Reliability*, 2015.

[45] C. Qi, L. Xiao, T. Wang, J. Li, and L. Li, "A highly reliable memory cell design combined with layout-level approach to tolerant single-event upsets," *IEEE Transactions on Device and Materials Reliability*, 2016.

[46] J. Guo, L. Zhu, W. Liu, H. Huang, S. Liu, T. Wang, L. Xiao, and Z. Mao, "Novel radiation-hardened-by-design (rhbd) 12t memory cell for aerospace applications in nanoscale cmos technology," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 2017.

[47] B. S. Gill, C. Papachristou, and F. G. Wolff, "Interactive presentation: A new asymmetric sram cell to reduce soft errors and leakage power in fpga," in *Conference on Design, automation and test in Europe (DATE'07)*. EDA Consortium, 2007.

[48] S. Miao, P. Ou, X. Zhou, and L. Wang, "Zero-hardened sram cells to improve soft error tolerance in fpga," in *International Symposium on Intelligent Information Technology Application (IITA'08)*. IEEE, 2008.

[49] J. Guo, L. Xiao, and Z. Mao, "Novel low-power and highly reliable radiation hardened memory cell for 65 nm cmos technology," *IEEE Transactions on Circuits and Systems I: Regular Papers*, 2014.

[50] N. Azizi, F. N. Najm, and A. Moshovos, "Low-leakage asymmetric-cell sram," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 2003.