HW/SW Co-design for Reliable In-memory Brain-Inspired Hyperdimensional Computing

Simon Thomann and Paul R. Genssler, Member, IEEE and Hussam Amrouch, Member, IEEE

Abstract—Brain-inspired hyperdimensional computing (HDC) is continuously gaining remarkable attention. It is a promising alternative to traditional machine-learning approaches due to its ability to learn from little data, lightweight implementation, and resiliency against errors. However, HDC is overwhelmingly data-centric similar to traditional machine-learning algorithms. In-memory computing is rapidly emerging to overcome the von Neumann bottleneck by eliminating data movements between compute and storage units. In this work, we investigate and model the impact of imprecise in-memory computing hardware on the inference accuracy of HDC. Our modeling is based on 14 nm FinFET technology fully calibrated with Intel measurement data. We accurately model, for the first time, the voltage-dependent error probability in SRAM-based and FeFET-based in-memory computing. Thanks to HDC’s resiliency against errors, the complexity of the underlying hardware can be reduced, providing large energy savings of up to 6x. Experimental results for SRAM reveal that variability-induced errors have a probability of up to 39%. Despite such a high error probability, the inference accuracy is only marginally impacted. This opens doors to explore new tradeoffs. We also demonstrate that the resiliency against errors is application-dependent. In addition, we investigate the robustness of HDC against errors when the underlying in-memory hardware is realized using emerging non-volatile FeFET devices instead of mature CMOS-based SRAMs. We demonstrate that inference accuracy does remain high despite the larger error probability, while large area and power savings can be obtained. All in all, HW/SW co-design is the key for efficient yet reliable in-memory hyperdimensional computing for both conventional CMOS technology and upcoming emerging technologies.

Index Terms—Reliability, In-memory Computing, Hyperdimensional Computing.

1 INTRODUCTION

In recent years, machine-learning models based on traditional algorithms like Deep Neural Networks (DNNs) have made steady progress. Such advancements are often associated with larger, more complex neural networks, further increasing their already large demand for processing power and memory. The traditional von Neumann architecture is reaching its limits for such overwhelming data-centric applications. As a matter of fact, data movement between memory and compute units profoundly contributes to the total energy consumption [41] and does form the key bottleneck. Furthermore, the significant amount of training data and the iterative training concept to tune the model’s weights both exacerbate the energy and processing-power challenges. Dedicated hardware accelerators, like Google’s Tensor Processing Units (TPUs) [15], aim at accelerating NN inference and training with large power-hungry on-chip memories. In such accelerators, the data is moved from external memory to the on-chip memories once and then repeatedly feed to the systolic MAC arrays reducing off-chip data movement. Emerging non-volatile memories consume less energy but are not as reliable as conventional CMOS-based SRAMs. However, DNNs are very susceptible to bit-errors in which a single bit flip can drop the inference accuracy almost to zero [13].

Brain-inspired Hyper-Dimensional Computing (HDC) addresses the challenges of costly iterative training and the need for large amounts of training data. Similar to the human brain, patterns can be learned in one iteration from little training data. For hand gesture recognition, a few samples are sufficient [7]. To create such an HDC model, only a single pass over the data is sufficient. This capability is called one-shot learning and avoids the costly iterative training required by DNNs. In addition, HDC operations, unlike in DNNs, do not rely on expensive floating-point matrix multiplications during training but are lightweight and highly-parallelizable bitwise operations. The same operations are furthermore used for both training and inference, avoiding two distinct hardware implementations. All of this is achieved through the use of vectors with thousands of independent components—hypervectors. Hence, an individual component has little impact on the overall accuracy resulting in high robustness against noise in the underlying memory as well as a high resiliency against errors in the performed operations [19], [28]. HDC has been successfully applied to a wide range of application domains such as language recognition [16], gesture recognition [35], seizure detection [7], DNA pattern matching [21], image classification [22], and more. In all applications, the inference operation associates an unlabeled query hypervector with the previously-trained class hypervectors by computing a similarity metric between them (e.g., Hamming distance). The computation is done in the Associative Memory (AM), which stores the class hypervectors. With traditional computer architectures, the processing unit has to compute the similarity with each class. Because of the hypervectors large size (e.g., 10 kbits), data movement is again a bottleneck in existing von Neumann architectures when employed to implement HDC, especially in regards to applications with a large number of classes.
**Novel In-memory Computing** architectures address this bottleneck by implementing the similarity computation directly within the memory where the data resides. The class hypervectors are not stored in regular memory but in a Ternary Content Addressable Memory (TCAM) [33]. In the traditional address-based memory concept, an address is provided to retrieve data. In a TCAM, the data is provided, and then its address is returned if the data exists. This approach can also be used for inference in HDC. If the unknown query hypervector is applied as “data” to the TCAM, the result is not the address but the similarities to each class. Therefore, the inference can be fully parallelized, accelerating this step. In addition, applying the query hypervector is the only data movement reducing the energy costs significantly. However, the in-memory operations are less precise due to their analog implementation. The quality of the peripheral circuitry and device-to-device process variation seriously impact the precision, especially in nano-scaled devices (e.g., 14 nm nodes and below). This holds even more when it comes to emerging memory technologies. Emerging non-volatile memories (NVMs), like Spin-Transfer Torque Magnetic RAM (STT-MRAM) [20], Resistive RAM (ReRAM) [1] or Ferroelectric FET (FeFET) [4], [9], promise an increased energy and area efficiency over conventional SRAM technology. Their single-device design require less die area compared to traditional 6T or 8T SRAM cells [37]. Furthermore, emerging NVMs consume significantly less power since they are non-volatile and can be turned off without loss of data [37]. However, these technologies are not yet as mature as CMOS-based SRAM and are therefore more affected by variation [24], [32], [39]. Nevertheless, HDC is inherently resilient against errors. Hence, the software can tolerate imperfect yet more efficient hardware, and HW/SW co-design becomes essential.

**Our key focus** in this paper is in-memory brain-inspired hyperdimensional computing. Based on 14 nm FinFET model fully calibrated with Intel measurements for both transistor characteristics and variability, we implement an SRAM-based TCAM to compute the similarity metric Hamming distance directly within the memory. The hypervector is divided into blocks and mapped to individual TCAM arrays. The computation is less precise in larger blocks and is additionally impacted by process variation. Thus, we model the probability of error in the Hamming distance computation based on block size and operating voltage in order to explore the available design space and the existing tradeoffs. Independent of the hardware, at the software level, we model the impact of imprecise distance computations on the inference accuracy for different block sizes. Finally, we combine both models into a framework to investigate the reliability of in-memory hyperdimensional computing. Based on this, we explore various HW/SW co-design tradeoffs. Figure 1 provides a general overview of our work starting from the transistor level to circuit level all the way up to the software level and the final HW/SW co-design. With a given set of application-depended class and query hypervectors, our flow enables the exploration of various tradeoffs. We explore two different applications (language classification and image classification), showing how software matters and HW/SW co-design is key. As shown, our framework has well-defined interfaces to accommodate different circuit-level models for the similarity metric computation. Therefore, our work is not limited to a certain technology node. Other technologies can be integrated within our framework as long as they offer energy, computation latency, and error probability models. In this work, we investigate not only conventional CMOS-based SRAM, but also emerging NVM-based Ferroelectric FinFET (Fe-FinFET). The latter promises a higher area and energy efficiency at the cost of a higher probability of error. To ensure fair comparisons, our Fe-FinFET device model is based on the same 14 nm FinFET baseline and is subject to the same amount of variations.

**Existing works** lack several key aspects. In [14], a large monolithic array of 45 nm SRAM-based XOR gates and counters has been proposed, whereas we utilize advanced 14 nm FinFET TCAMs. In combination with the fact that process variation was not considered in [14]; hence, no errors could be modeled. We employ a calibrated transistor characteristics and variability model that allows us to correctly investigate the impact of process variation on the inference accuracy. In a proposed ReRAM-based crossbar, the process variation model is basic, and its impact is hidden by slower computation [14]. We demonstrate that in SRAM-based designs, process variation cannot be addressed similarly and instead requires a higher voltage increasing energy consumption. The tradeoff between energy and inference accuracy in [14] is based on a simplified assumption about the impact of errors in HDC. We show that the resiliency against errors of HDC is much larger. A FeFET-based TCAM design was proposed in [33]. They consider process variation...
but only for the older MOSFET design, whereas FinFET is much more susceptible to variation. In contrast to our work they do not evaluate the impact on the inference accuracy at software level.

**Distinction from existing work:** None of the existing works consider detailed cross-layer modeling starting from the transistor level all the way up to the software level in a holistic way. We demonstrate that HW/SW co-design is the key to efficient yet reliable in-memory HDC computing.

**Our novel contributions within this paper are as follows:**

1. At the software level, we investigate how imprecise similarity metrics computations impact the inference accuracy of different HDC models. We divide the hypervectors into small blocks and limit the largest detectable Hamming distance.
2. At the hardware level, we model the energy consumption as well as the probability of errors in the underlying SRAM-based TCAM caused by block size, process variation, and voltage. For accurate modeling, we employ 14nm FinFET technology fully calibrated with Intel measurement data. Further, we extend our model to additionally account for the Fe-FinFET technology. This allows, for the first time, a fair comparisons of the reliability of in-memory brain-inspired HDC when implemented with conventional SRAMs vs. emerging NVMs.
3. Based on (1) and (2), our HW/SW co-design explores the impact of error-prone TCAM-based similarity computations on the inference accuracy for the first time. It reveals that HW/SW co-design opens doors to eliminate hardware errors and keep variability effects, which are very challenging in nano-scaled technologies, at bay. We further explore the tradeoff between energy and inference accuracy under process variation, demonstrating energy savings of 6x through voltage reduction is possible despite the high induced probability of error of up to 39%. We demonstrate that Fe-FinFET has higher losses in the inference accuracy and propose a replica technique to mitigate those losses to the level of SRAM.

### 2 Hyperdimensional Computing

HDC has been recently researched in a wide range of application domains [7], [16], [21], [22], [35]. The concept itself was proposed by Kanerva in 2009 [16]. The technique can detect patterns and classify data by mapping real-world entities into HD space. Such a space is created by placing barely changes the represented real-world value. In contrast, numbers in traditional binary representation change drastically if a single significant bit is flip. Thus, HDC is intrinsically robust against noise and resilient to faults [19], [28]. Due to the very high dimensionality, the likelihood that two hypervectors are orthogonal is very large. Two binary hypervectors are orthogonal if their normalized Hamming distance is approximately 0.5. This metric is determined by the number of individual bit pairs that are different between both hypervectors. In other words, the number of ones in their XOR product. The resulting number is divided by the length of the hypervectors to normalize it. For other types of hypervectors, the operation (measure similarity) is determined differently but retains its meaning.

Using the generated hypervectors, the complex real-world entity can be encoded into HD space with the other three basic operations: addition $\oplus$, multiplication $\otimes$, and permutation $\Pi$. The addition $\oplus$ bundles two or more hypervectors into a single hypervector of the same dimension. The multiplication $\otimes$ binds two or more hypervectors together. The third operation, permutation $\Pi$, operates on a single hypervector and makes it orthogonal to itself.

#### 2.1 Language Recognition with HDC

To detect the language of an unknown text, not only are the letters’ frequencies important but also typical combinations of them, like “-ing” or “-tion” in English. The $n$-gram data structure is able to capture both aspects by encoding $n$ individual letters $L_i$ and their relation to each other through permutation [36]. The letter hypervectors $HV_A$ to $HV_Z$ are created randomly once, remain unchanged, and are stored in the Item Memory (IM).

$$HV_{n\text{-gram}} = HV_{L_0} \otimes \Pi^1(HV_{L_1}) \otimes \ldots \otimes \Pi^{n-1}(HV_{L_{n-1}})$$

With a sliding window of size $n$, the whole text is encoded and the $n$-grams are added together to form a single hypervector. To train the HD model, for each language (class) a reference text is encoded. The class hypervector to language mappings are stored in the AM, the HD model. During inference, the unknown text is encoded with the same $n$-gram technique and the same hypervectors from the IM. The encoded hypervector is presented to the AM as a query. The similarities to all stored class hypervectors are computed. The label (i.e., language) of the class hypervector with the highest similarity is returned as the classification result.

In this work, we focus on binary hypervectors. The $n$-gram encoding is done with $n = 4$ as it yields the highest inference accuracy [36]. Eight different European languages are each trained with 1 million characters from the Wortschatz Corpora [34]. For inference, a query for each language is created with 1000 sentences from the Europarl Parallel Corpus [23]. The dependency of the inference accuracy on dimension is shown in Figure 2. Increasing the dimension beyond 10,000 would benefit the accuracy only marginally. Hence, we limit the investigation to 10,000-bit hypervectors.
2.2 Image Classification with HDC

The MNIST dataset is a common benchmark and includes a total of 70,000 labeled images [25] for the ten digits from zero to nine. The images have a uniform size of 28×28 pixels. For each of the 784 pixel positions, a random hypervector is generated. The gray-scale images are binarized, and the position hypervectors of all white pixels are bundled into a single image hypervector. Per class, 6000 image hypervectors are selected for training and bundled into a class hypervector. No further processing, like retraining, is done. The AM contains ten classes (from digit 0 to digit 9) and is queried with the remaining 10,000 image hypervectors.

3 Hardware-Level Analysis and Modeling

In typical memory, an address is provided to return data. In Content Addressable Memory (CAM), data is provided and, if the data is already present in the memory, its address is returned. A TCAM is an extension and allows a lookup with “don’t care”. This lookup capability makes TCAMs perform best in search operations. While in typical memory, the data at each address would have to be compared sequentially, in a TCAM the lookup is parallelized. Hence, latencies within a few clock cycles can be achieved. TCAMs are therefore already used in ultra-high performance applications like network switching [17] or search engine accelerators in databases [43]. The third “don’t care” state can be used, for instance, to perform masked IP address lookups [8]. In HDC, the AM can be implemented efficiently with a TCAM avoiding costly data movement. The query hypervector represents the data to search for. Instead of address lookup, the Hamming distance to all stored class hypervectors can be computed in parallel. Therefore, instead of “search” it is called “comparison” in the rest of the paper.

In this section, we first introduce our calibrated CMOS transistor model that includes process variation. Second, we describe the SRAM-based TCAM cell design and, based on it, the block structure to store and compare a part of a hypervector. Then we use the process variation data from our transistor model to derive the error probabilities for each Hamming distance depending on voltage and block size.

3.1 Technology Modeling: 14 nm FinFET Calibration

In this work, we reproduce Intel’s 14 nm FinFET measurements [31] for their mature high-volume manufacturing process. With SPICE simulations, we carefully tune the transistor model-card parameters for the the industry-standard compact model of FinFET (BSIM-CMG) until they are in excellent agreement with the measurements, both for nFinFET and pFinFET, as demonstrated in Figures 3(a) and 3(b). This applies to both, the \( I_D-V_G \) and \( I_D-V_D \) transistor properties. In a second step, we calibrate the model against the measurements for device-to-device variability. All important sources of manufacturing variability (gate work function, channel length, fin height, fin thickness, and effective oxide thickness) are modeled for a comprehensive representation of the process variation. Based on the calibrated compact industrial model presented first and Monte-Carlo SPICE simulations, we calibrate the standard deviations for each mentioned source of variability.

3.2 Single TCAM Cell

A single TCAM cell with a 16 CMOS-transistor design couples two SRAMs S1 and S2 as shown in Figure 4(a) [33]. The cell’s data \( C \), a single bit of a class hypervector, is written to the two SRAMs in a complementary fashion. For instance, when \( C = 1 \), S1 and S2 are in the logical 1 and 0 state. A single SRAM is a bi-stable element formed by an inverter loop, the labeled nodes (‘L’ and ‘R’) are on the negated side. The left SRAM S1 holds 1, but its negated ‘L’ node expresses the inverted value 0. For a lookup, the Match Line (ML) is pre-charged. Then the query data \( Q \) is applied to the Select Line (SL) (corresponds to the left SRAM S1) and inverted to SLB (corresponds to the right SRAM S2). If it is a match \( (C = Q) \), then the inverted ‘L’ and SL are complementary \((C \neq Q)\) and no conductive path from GND to the ML is formed. The TCAM cell is OFF because the voltage stays high. The same logic applies analogously to the right-hand side for the non-inverted ‘R’ and inverted SLB \((C \neq Q)\). If it is a miss, either ‘L’ and SL or ‘R’ and SLB are active at the same time, their associated transistors form a conductive path and discharge the ML. The TCAM cell is ON.

3.3 Our SRAM-based TCAM Array

Each individual TCAM cell stores a single bit of a class hypervector. Multiple cells are combined into a block to represent multiple bits of the hypervector. All TCAM cells in
the block share the same periphery and access logic as shown in Figure 4(c). This includes the Bit Line (BL) and Word Line (WL) to write the class hypervector. In our model, this write operation occurs only during an initialization phase and thus is not within the focus of our framework. Hence, we exclude it from the evaluation and also simplify the SPICE circuit implementation by replacing the BL and WL with individual voltage sources.

The ML is shared with all TCAM cells within one block and an integral part of the TCAM block design. Before an inference operation, the ML is pre-charged. Then the query hypervector is applied to the block. Recall that each block only represents a few bits of the class hypervector, and thus only the respective bits of the query hypervector are applied. In each individual TCAM cell, the applied bit and the stored bit are compared as described in Section 3.2. If the result is a miss, the TCAM cell will establish a conductive path from the ML to GND and discharge the ML. The more misses occur, the more conductive paths are established, the faster the ML discharges due to the smaller total resistance. In other words, the discharge rate reflects the number of cells reporting a miss. The number of misses between the stored class hypervector and the applied query hypervector is equal to the Hamming distance. Hence, in practice, the TCAM block realizes the Hamming distance computation.

In our SPICE simulations, the ML is connected to a clocked self-referenced sense amplifier (CSRSA) [33]. Its schematic is illustrated in Figure 4(b), and it converts the discharge rate from the voltage domain to the temporal domain. The operation latency determines the Hamming distance of this block. An example is shown in Figure 4(d) for a block size of 15 and Hamming distances from one to seven bits. Similar to [14], we observe a linear dependency between the number of misses and the discharge rate. While one and two misses are clearly separated by 0.01 ns, this gap roughly halves between two and three misses. The size of the gaps also depends on the size of the block, since large blocks have higher parasitic capacities increasing the discharge time for any number of misses. Nevertheless, separating large Hamming distances is challenging and impacted by block size and voltage. Determining the largest detectable distance and attributing a cost in terms of chip area, additional operation latency, or energy is implementation-specific. Models can be provided for our framework to include such costs into the analysis, extending the explored space.

In this work, we consider TCAM blocks from a size from two bit up to 25 bits. At this upper limit, our evaluation points to a low inference accuracy in conjunction with the maximum precision of 7 bits. Hence, any larger block sizes would result in low-accuracy systems. For a full AM (e.g., 10,000 bits), multiple block instances are required to store a full hypervector.

3.3.1 TCAM Block Energy Consumption

The block size affects energy consumption. Larger blocks naturally require more energy but their TCAM cells all share the same periphery; in our circuit, the CSRSA. Hence, their overhead per bit is smaller. The energy consumption of a single block is accurately extracted from the SPICE simulations. All currents over the operation latency of the complete similarity computation are integrated. The operation latency is the time between the flank of the CSRSA’s enable signal and the output voltage dropping below 50 % VDD. It is noteworthy that our framework is modular, and hence any other alternative latency and/or energy models can be included. In the targeted 14 nm FinFET technology, the nominal voltage is 0.7 V. To explore existing tradeoffs between reliability, operation latency, and energy under the effect of voltage, we study a wide range of operating voltages starting from 0.5 V up to 1.0 V.

3.4 Modeling the Error Probability

Variation is inherent to every manufacturing process. It impacts key electrical characteristics of the transistors and hence the reliability and speed of a TCAM cell. Some cells establish a stronger conductive path from the ML to GND. The discharge of the ML is an analog process without a clock enable signal. Thus, variation directly impacts key electrical characteristics of the transistors and can cause errors.

Our calibrated process variation model (see Figures 3(c) and 3(d), described in Section 3.1, is applied to the underlying transistors of a TCAM cell. We conduct 1000 Monte-Carlo SPICE simulations per block size and all possible Hamming distances. The variation-free nominal operation latencies are shown in Figure 4(d), whereas the results of the Monte-Carlo SPICE simulations are presented as histograms in Figure 5 for three different voltages. We analytically model the operation latency distribution for each Hamming distance and build our probabilistic error model. Without process
variation, our framework then maps the nominal operation latency to a Hamming distance (Figure 4(d)). With process variation, the framework samples the operation latency distribution (Figure 5) of the nominal Hamming distance. If the variation from the nominal operation latency is larger than half the distance to the neighboring Hamming distance, then this neighboring Hamming distance is reported as the result of the computation. Since this is the incorrect distance, it is counted as an error. As it can be seen from Figure 5, most incorrect Hamming distances are within one bit of the nominal distance. Only for large distances (i.e., small latencies), higher deviations are possible. The error probabilities for the block size of 15 bits are shown in Figure 6. The maximum detectable Hamming distance is defined as seven bits; any shorter operation latency is also reported as seven bits. While three bits, due to variation, can be incorrectly reported as two or four bits, seven bits cannot be reported as eight bits because of the definition of a maximum detectable Hamming distance. Thus, the error probability for seven bits is lower.

### 3.5 Impact of Operating Voltage

A higher supply voltage $V_{DD}$ increases the gate voltage at the two transistors on the discharge path. Thus, the discharge rate is higher and the increase of $I_{ON}$ dominates the linear increase of charge $Q$ of the ML. The increase of $V_{DD}$ also reduces the spread of the individual Hamming distance groups. As shown in Figure 6, this directly influences the error probability. Contrary to the expectations, 0.5 V performs better than 0.7 V. A detailed analysis with finer voltage steps reveals that the interaction between operation latency and spread of the distributions benefits 0.5 V. The lower voltage causes wider distributions but those are farther apart. With 0.7 V, the distributions are narrower but closer together, resulting in higher overlap and a higher probability of error. The drawback of the high 1.0 V level is higher energy consumption. Even though the discharge rate is higher, the number of charge carriers that have to be moved increases as well. This increase is directly proportional to the energy and outweighs other reductions.

However, HDC’s resiliency against errors creates the opportunity for energy savings. If a higher error probability can be tolerated by the HDC model, than a voltage reduction also reduces the energy consumption. At 0.5 V, a 15-bit block consumes 0.73 fJ per similarity computation in contrast to 4.53 fJ at 1.0 V. The numbers are derived from our SPICE simulations as discussed in Section 3.3.1. With our HW/SW co-design framework, the tradeoff between inference accuracy and energy consumption is investigated in Section 5.

### 4 Software-level Analysis and Modeling

At the software level, the inference accuracy for an HDC model is determined. Our framework only requires the encoded class and query hypervectors. Therefore, it is agnostic to the actual application and the encoding of the hypervectors. Currently, only binary hypervectors are investigated since the underlying hardware models only supports Hamming distance computation.

Traditional SRAM arrays used in processor caches are not wide enough to hold a large hypervector [18], [40]. Additionally, compute-centric architectures only support fixed-length operations like a 64-bit XOR. Hence, multiple operations have to be executed to compute the Hamming distance between two hypervectors. If a given distance threshold has already been passed before the full hypervectors are processed, the computation can stop earlier. Such a simplification can reduce energy and the compute time, especially in embedded systems. We introduce this concept as the precision of a block. For traditional hardware architectures, a block is as big as the whole hypervector and the precision is the distance threshold. While this concept is optional for such architectures, it is required with TCAM arrays.

In-memory Hamming distance computation based on TCAMs/CAMs is restricted to a maximum vector length as shown by our TCAM block design in Figure 4 and others [14], [33]. Such a block could be, for example, a TCAM array or an ReRAM crossbar [14]. One reason to limit the size of the block is the challenge for the hardware to differentiate between different Hamming distances as discussed in Section 3. In case of TCAM cell designs, the higher Hamming distances overlap significantly, making their separation difficult. The concept of precision can be applied again. It is a simplification that aims to reduce the implementation effort of a block by limiting the reported Hamming distance. As an example, the
15-bit block in Figure 4(d) only reports correctly until seven bits; any higher difference is also reported as seven.

Our software-level analysis is orthogonal to the underlying hardware and relies on models. Initially, a model of error-free and variation-free hardware is assumed. It accepts a partial class and query hypervector and always computes the correct Hamming distance. This computation is repeated for all parts of the class and query hypervector and all classes. The class with the lowest Hamming distance is selected as the inference result. The result is compared with the true label associated with the test hypervector. If the inferred class and the true label match, then the inference is correct. The overall inference accuracy is the ratio of correctly classified test samples to the total number of test samples. Based on the assumption of an ideal hardware implementation of a block, the inference accuracy is as high as possible for the given HDC model.

In Figure 7, different block sizes and precision are shown with the examples of language recognition and image classification. As a general observation, the inference accuracy starts to drop significantly if the precision of a block is limited to less than half of the block size. This is to be expected since information is lost if the Hamming distance is larger. A small distance in another block cannot compensate for that. Interestingly, in the image classification application, the inference accuracy does not drop as fast. The reason is the already high overlap of the class hypervector with each other. On average, their normalized Hamming distance is 0.21 in contrast to 0.44 for languages. In other words, the diversity in the image classification model is lower. Consequently, most queries are similar to the class hypervector and have a lower Hamming distances. Precision does not limit as much. This suggests the possibility of a precision-aware encoding scheme to increase the similarities of the classes while at the same time retaining high inference accuracy.

5 HW/SW Co-design and Evaluation

For a holistic analysis of an in-memory HDC system, models at hardware and software level have to be integrated. In this section, we demonstrate the promise of HW/SW co-design to obtain an efficient design solution. The error properties of the underlying hardware are jointly considered with the error resiliency of the target application at the software level.

5.1 Experimental Setup

At the transistor level, we employ our 14 nm FinFET model described in Section 3.1. The TCAM cell design and the TCAM block circuit is described in Section 3.2 and Section 3.3, respectively. The error model is developed in Section 3.4. We evaluate the inference accuracy with five applications in total. Language recognition and image classification are introduced in Section 2. To confirm our analysis and conclusion, we additionally evaluate EMG gesture detection [27], voice recognition ISOLET [10], and heart disease detection CARDIO [10]. The design space includes the voltage levels 0.5 V, 0.7 V, 0.8 V, and 1.0 V, block sizes from two to 25 bits, and an application-dependent subset of dimensions from 2000 bits to 10,000 bits. The hypervectors are partitioned and assigned to the TCAM blocks. The similarity metrics are computed per block and provided to the error model to capture the impact of process variation. Additionally, the energy consumption of each TCAM block, which depends on the Hamming distance, is aggregated. The inference accuracy and energy consumption form one point in the design space.

5.2 Impact of Variability-induced Errors

As discussed in Section 3.1, process variation is a major concern in advanced technologies such as FinFET. This is reflected in the probability of errors for a Hamming distance computation of close to 40% at the nominal voltage of 0.7 V (Figure 6). However, the inference accuracy decreases at most by 0.45% at 0.7 V (0.28% on average) as shown in Figure 8. We have demonstrated in Figure 6 that a reduced voltage of 0.5 V actually reduces the probability of errors. This is reflected by an increased inference accuracy. To further lower the probability of errors, the voltage has been increased to 1.0 V resulting in a maximum loss of inference accuracy by 0.21% (0.11% on average), close to the fluctuations caused by the inherent randomness of any HDC model. These results show that HDC is also resilient against errors in the computation of the similarity metrics. Although the impact of process variation has to be considered, it does not dominate the tradeoffs available to a system designer.

5.3 Tradeoff Energy vs. Accuracy

Figure 9 shows the Pareto fronts in the design space for all five applications. The impact of the application on the HW/SW co-design is discussed in the next section.

Voltage influences the impact of process variation on the inference accuracy as discussed in Section 5.2. Our HW/SW
co-design shows that the increased energy consumption with 1.0 V does not outweigh the accuracy gain for almost all configurations. A higher voltage is only Pareto-optimal if a permille in inference accuracy is required. However, due to the random nature of any HDC model, such small differences are overshadowed by a “lucky”, or “unlucky” initialization of the random item hypervectors. Hence, an increased voltage does not offer a significant benefit and instead offers the possibility to reduce energy consumption. Accordingly, the large majority of Pareto-optimal configurations use 0.5 V.

**Block size** in relation to precision creates an upper bound on the inference accuracy for Pareto-optimal configurations. For example, to limit the accuracy loss to 2.5 %, the precision has to be at least half of the block size. A special case is a precision of one bit transforming the TCAM block into a comparator. Either all bits are equal (Hamming distance is zero) or all are different (Hamming distance is the block size). To maintain some accuracy, the only viable block size is two. With 10,000-bit hypervectors, the loss in inference accuracy is limited to 0.5 %. Indeed, for all Pareto-optimal configurations, losses smaller than 0.5 % require a high level of precision and small block sizes. In other words, full precision and little to no errors in the similarity metrics computation. However, process variation is still a source of errors, which have to be tolerated by the HDC model.

The **dimension** of the hypervectors creates a similar bound on the inference accuracy. The lowest energy consumption is achieved with 2000-bit hypervectors. For language recognition, about a quarter of the Pareto-optimal configurations use this dimension but do not exceed 96 % accuracy. A similar portion of the configurations use 9000-bit hypervectors and thus achieve the highest accuracy. This shows that the inherent resiliency against errors is not purely dependent on a higher dimension as a redundancy buffer.

### 5.4 Impact of Application on HW/SW Co-design

In Section 4, the similarities of the class hypervectors to each other are discussed for two applications. The ten class hypervectors of the image dataset are more similar to each other, whereas the class hypervectors in the language classification model are almost orthogonal. This difference impacts the resiliency against errors. The more similar class hypervectors in the image classification make the HDC model more resilient, as shown in Figure 7. The higher resiliency enables more potential for energy savings. Therefore, block sizes of up to 25 are Pareto-optimal for image classification. Additionally, the impact of process variation is smaller compared to language recognition. The reason is the smaller Hamming distances of query and class hypervectors. As shown in Figure 6, the probability of errors is smaller for lower distances. Furthermore, the highest Pareto-optimal dimension is 8000 bits for images compared to 10,000 bits for languages. Pareto-optimal configurations for both applications only agree in 0.5 V and require different parameters otherwise. These differences highlight the importance of HW/SW co-design. If a 0.5 % loss in inference accuracy is acceptable, an efficient implementation of an image classifier saves 4.5x energy, whereas 11.5x is possible for languages.

### 5.5 Comparison to State of the Art

In [14], a 4-bit ReRAM crossbar is proposed for in-memory Hamming distance computation. Similar to our work with SRAM-based TCAM arrays, the hypervector is partitioned and mapped to the hardware blocks. In contrast to our approach in which HW/SW co-design is employed to eliminate variability-induced errors, they nullify the impact of process variation by increasing the operation latency. Therefore, errors only occur if they scale down the voltage of their circuit to reduce energy consumption. Due to the reduced voltage, the result of all Hamming distance computations is increased by one bit. For example, a distance of two bits is always reported as a distance of three bits. If the Hamming distance is four bits, five would have to be reported due to the shift. However, the 4-bit ReRAM crossbar can only report distances of up to four bits. We implement their error model in our framework. Thanks to its modularity, we can quickly repeat the analysis analogously and evaluate their error model for different dimensions and applications. In their evaluation, the ReRAM crossbar array section that operates at a reduced voltage always causes an error of one bit in the Hamming distance. With the number of scaled voltage ReRAM crossbars (i.e., number of bit errors), they assigned an inference loss determined by a theoretical analysis of one application. In contrast, our framework evaluates the error models by executing the HDC models. We show that the ReRAM error model does perform better than previously reported. Their assumption, that each scaled voltage ReRAM crossbar causes one bit error, is too pessimistic because the shift applies to all crossbars and thus cancels out. Instead of 4.0 % if all crossbars used a scaled voltage [14], the accuracy loss amounts to 0.2 %, which is comparable to the SRAM-based TCAM model for a dimension of 10,000 bits. The accuracy loss for other dimensions is comparable as well. For image classification, the impact of both error models on the inference accuracy is similar.

### 6 HDC WITH EMERGING NVM TECHNOLOGIES

Data-intensive workloads like big data and deep neural networks demand for larger on-chip memories. However, traditional CMOS-based SRAM consumes lots of power and chip area. With a slow-down in classical CMOS technology scaling, emerging NVM technologies gather more and more
Fig. 10. (a) FeFET where the high-$\kappa$ layer is replaced by a thick (8 nm to 10 nm) layer of ferroelectric material (Hf$_{0.5}$Zr$_{0.5}$O$_2$). (b) Polarizing the ferroelectric layer in the gate stack creates two distinguishable states, i.e., low $V_{th}$, and high $V_{th}$, which correspond to high and low current, respectively.

FeFET attracts interest in academia and industry alike. GlobalFoundries has already demonstrated various prototypes [5], [9] and Intel has recently reported an endurance breakthrough of $10^{12}$ cycles [3]. Furthermore, only two FeFETs are necessary to implement a TCAM cell as shown in Figure 4(a) making a FeFET-based TCAM array 8X denser than an SRAM implementation [44] creating new tradeoffs.

6.1 Fe-FinFET-based TCAM Array

Previous works investigated planar MOSFET-based FeFETs, which are less susceptible to variation. In our implementation, we employ the Intel 14 nm FinFET model, also used in the SRAM-based design, as a base. To create a Fe-FinFET, the FE layer is added. Thus, the underlying FinFET base is complemented by a state-of-the-art ferro transistor physics-based compact model [33]. Since we employ the same calibrated model described in Section 3.1 for the underlying FinFET, we can study and compare the impact of process variation on Fe-FinFET and SRAM in a fair way for the first time. Our transistor level operation (initializing the polarization, write transistor into desired state) of the Fe-FinFETs is similar to recent state-of-the-art shown in [42]. To program the Fe-FinFETs, a 4 V pulse is applied for 10 µs to saturate the polarization for the best read performance. The state of the Fe-FinFETs changes rarely and only if the class hypervectors have to be updated. Hence, write latency and endurance are not of concern.

In contrast to the 16 CMOS transistor design, a Fe-FinFET-based TCAM only requires two Fe-FinFETs as depicted in Figure 11. Both designs use the complementary storing scheme described in Section 3.2. If a logical 0 has been stored in the TCAM, then Fe-FinFET$_i$ is in the low-$V_{th}$ state (green). A logical 1 on the SL triggers the discharge and thus signals a miss. The structure of a full array is analogous to the design shown in Figure 4(c). Only the SRAM-based TCAM cells are replace with Fe-FinFET-based TCAM cells.

6.2 HDC with Fe-FinFET Under Process Variation

To study an AM with a Fe-FinFET-based TCAM array, the methodology described in Section 3.3 is applied again. Comparability is ensured by applying the same capacity to the ML for both designs, Fe-FinFET and SRAM. The distributions of the operation latencies are shown in Figure 12. First, the voltage has a much higher effect on the latency compared to SRAM. Fe-FinFET is overall slower with up to 5.0 ns with one miss at 0.5 V. At 1.0 V, the latency is comparable to SRAM at 0.5 V. Second, the variation is higher, clearly visible by the large overlap of the latencies representing different Hamming distances. Similar to SRAM, higher voltages do reduce the overlap. At 0.5 V, the latencies for a one-bit Hamming distance spread from 0.8 ns to 5.0 ns. The sizeable overall overlap is due to Fe-FinFET’s higher susceptibility to process variation compared to SRAM. This is expected since SRAM is a very mature technology in contrast to Fe-FinFET, which is still in the prototype stage.

Figure 13 shows that the higher process variation results in a higher error probability. Whereas SRAM peaked at 40 %, Fe-FinFET reaches 78 %. Interestingly, the observed trend from Figure 6 is not repeated. Instead, an increase in supply voltage proportionally decreases the error probability. The
worse error probabilities translate further into higher inference accuracy losses. Although HDC is very robust against noise, such high error rates inevitably reduce the inference accuracy. Figure 14 shows this increased loss. For seven-bit block sizes, while in SRAM, the loss in inference accuracy can be limited to less than 0.05% by using higher voltages, Fe-FinFET can achieve 0.3% at best. For smaller block sizes, inference accuracy loss can be lower but not as low as with SRAM. Energy for a comparison operation in the AM (see Figure 15). Nevertheless, the static power consumption in a Fe-FinFET-based system is much lower since it can be turned off due to its non-volatility. To evaluate those savings and compute the total energy consumption, the ratio of comparison operation to idle time has to be known, which highly depends on the target system.

However, such a scheme does not scale if the area budget is sufficient to accommodate a large SRAM-based AM. As shown in Figure 2, increasing the vector length has diminishing returns. To still exploit the area savings, the Fe-FinFET-based arrays can be replicated multiple times. The query is applied to all arrays simultaneously, thus not increasing latency. Since all TCAM cells in each array are affected by process variation differently, the arrays can return different Hamming distances. The median distance is select, effectively ignoring the worst-affected and most inaccurate TCAM cells. In Figure 16, the benefits of this approach are shown. The given inference accuracy losses are averaged over all evaluated block sizes. With seven replicas, the loss in a Fe-FinFET-based system is on average 0.03% higher than in a SRAM-based one. Such a difference is negligible because the random initialization of the vectors can have a bigger impact. With three or five replicas, the difference is limited to at most 0.26% and 0.09%, respectively. Compared to the Fe-FinFET baseline, three replicas already reduce the inference accuracy loss by up to 0.29%. Further analyses have to be conducted to evaluate the increased power consumption.

In summary, for area or power-constrained systems, Fe-FinFET already offers an alternative to SRAM. If the area

### TABLE 1

|                          | SRAM | Fe-FinFET | Diff [%] |
|--------------------------|------|-----------|----------|
| **Transistors [#]**      | 16   | 2         | -800     |
| $E_{\text{max}}$ [\text{fJ}] | 1.15 | 1.24      | 7        |
| **Latency [\text{ns}]** | 0.099| 0.305     | 308      |
is not of concern, then Fe-FinFET-based TCAM arrays can be replicated to reduce the impact of variation. Advances in manufacturing technology and processes will also reduce this impact, leading to a more mature Fe-FinFET technology with better performance in HDC systems, among others.

7 Conclusion

In this work, we demonstrated that the resiliency of HDC against errors is larger than what has been previously assumed. We revealed the marginal impact of variability on the inference accuracy for SRAM-based TCAM cell and the role of applications. If a 0.5 % loss in inference accuracy is accepted, up to 11.5x energy saving would be possible in some applications. We also investigate the performance of Fe-FinFET-based TCAM cells and show the effect of process variation on such systems. Due to its prototype state, inference accuracy losses are higher but chip area and static power can be reduced. Such gains can be traded off to counteract the effects of process variation. All in all, HW/SW co-design is a key to achieve efficient, yet reliable in-memory hyperdimensional computing.

Acknowledgement

Authors thank X. Sharon Hu, Xunzhao Yin, and Kai Ni for their valuable support. This research was partially supported by Advantest as part of the Graduate School “Intelligent Methods for Test and Reliability” (GS-IMTR) at the University of Stuttgart and in part by the German Research Foundation (DFG) “ACROSS: Approximate Computing aCROSs the System Stack”.

References

[1] H. Akinaga and H. Shima, “Resistive random access memory (reram) based on metal oxides,” Proceedings of the IEEE, vol. 98, no. 12, pp. 2237–2251, Dec 2010.

[2] J. G. Alzate, U. Arslan, P. Bai, J. Brockman, Y. J. Chen, N. Das et al., “2 mb array-level demonstration of stt-mram process and performance towards l4 cache applications,” in 2019 IEEE International Electron Devices Meeting (IEDM), Dec 2019, pp. 2.4.1–2.4.4.

[3] W. Banerjee, I. V. Karpov, A. Agrawal, S. Kim, S. Lee, S. Lee et al., “Highly-stable (< 3% fluctuation) ag-based threshold switch with extreme-low off current of 0.1 pa, extreme-high selectivity of 10⁹ and high endurance of 10⁸ cycles,” in 2020 IEEE International Electron Devices Meeting (IEDM), 2020, pp. 28.4.1–28.4.4.

[4] T. S. Böske, J. Müller, D. Braunhaus, U. Schröder, and U. Böttger, “Ferroelectricity in hafnium oxide: Cmos compatible ferroelectric field effect transistors,” in 2011 International Electron Devices Meeting, Dec 2011, pp. 24.5.1–24.5.4.

[5] S. Beyer, S. Dünkel, M. Trentzsch, J. Müller, A. Hellmich, D. Utess et al., “Fefet: A versatile cmos compatible device with game-changing potential,” in 2020 IEEE International Memory Workshop (IMW), 2020, pp. 1–4.

[6] M. T. Bohr, R. S. Chau, T. Ghani, and K. Mistry, “The high-k solution,” IEEE Spectrum, vol. 44, no. 10, pp. 29–35, Oct 2007.

[7] A. Burrello, K. Schindler, L. Benini, and A. Rahimi, “One-shot learning for ieeeg seizure detection using end-to-end binary operations: Local binary patterns with hyperdimensional computing,” in 2018 IEEE Biomedical Circuits and Systems Conference (BioCAS), 2018, pp. 1–4.

[8] Y.-J. Chang, “A high-performance and energy-efficient team design for ip-address lookup,” IEEE TCAS-II, vol. 56, no. 6, pp. 479–483, 2009.

[9] S. Dünkel, M. Trentzsch, R. Richter, P. Moll, C. Fuchs, O. Gehring et al., “A fet based super-low-power ultra-fast embedded nvm technology for 22nm fdsoi and beyond,” in 2017 IEEE International Electron Devices Meeting (IEDM), Dec 2017, pp. 19.7.1–19.7.4.

[10] D. Dua and C. Graff, “UCI machine learning repository.” [Online]. Available: http://archive.ics.uci.edu/ml

[11] J. P. Duarte, S. Khandelwal, A. Medury, C. Hu, P. Kushwaha, H. Agrawal et al., “Bism-cmg: Standard finfet compact model for advanced circuit design,” in Conference 2015 - 41st European Solid-State Circuits Conference (ESSCIRC), 2015, pp. 196–201.

[12] W. J. Gallagher, E. Chien, T. Chiang, J. Huang, M. Shih, C. Y. Wang et al., “22nm stt-mram for reflow and automotive uses with high yield, reliability, and magnetic immunity and with performance and shielding options,” in 2019 IEEE International Electron Devices Meeting (IEDM), Dec 2019, pp. 2.7.1–2.7.4.

[13] M. A. Hanif, F. Khalid, R. V. W. Putra, S. Rehman, and M. Shafique, “Robust machine learning systems: Reliability and security for deep neural networks,” in 2018 IEEE 24th International Symposium on On-Line Testing And Robust System Design (IOLTS), 2018, pp. 257–260.

[14] M. Imani, A. Rahimi, D. Kong, T. Rosing, and J. M. Rabaey, “Exploring Hyperdimensional Associative Memory,” in 2017 IEEE International Symposium on High Performance Computer Architecture (HPCA), Feb 2017, pp. 445–459, ISSN: 2379-203X.

[15] N. P. Loippi, C. Young, N. Patil, D. Paterson, G. Agrawal, R. Bajwa et al., “In-datacenter performance analysis of a tensor processing unit,” SIGARCH Comput. Archit. News, vol. 45, no. 2, p. 1–12, Jun. 2017. [Online]. Available: https://doi.org/10.1145/3140659.3080246

[16] P. Kanerva, “Hyperdimensional Computing: An Introduction to Computing in Distributed Representation with High-Dimensional Random Vectors,” Cognitive Computation, vol. 1, no. 2, pp. 139–159, Jun. 2009. [Online]. Available: http://link.springer.com/10.1007/s12559-009-9009-8

[17] R. Karam, R. Puri, S. Ghosh, and S. Bhunia, “Emerging trends in design and applications of memory-based computing and content-
addressable memories,” *Proceedings of the IEEE*, vol. 103, no. 8, pp. 1311–1330, 2015.

[18] E. Karl, Z. Guo, J. Conary, J. Miller, Y. Ng, S. Nalam et al., “A 0.6 V, 1.5 ghz 84 mb sram in 14 nm finet cmos technology with capacitive charge-sharing write assist circuitry,” *IEEE Journal of Solid-State Circuits*, vol. 51, no. 1, pp. 222–229, 2016.

[19] G. Karunaratne, M. Le Gallo, G. Cherubini, L. Benini, A. Rahimi, and A. Sebastian, “In-memory hyperdimensional computing,” *Nature Electronics*, vol. 3, no. 6, pp. 327–337, Jun. 2020, number: 6 Publisher: Nature Publishing Group. [Online]. Available: https://www.nature.com/articles/s41928-020-0410-3

[20] A. Rahimi, P. Kanerva, and J. M. Rabaey, “A Robust and S. Natarajan, M. Agostinelli, S. Akbar, M. Bost, A. Bowonder, K. Lee, J. H. Bak, Y. J. Kim, C. K. Kim, A. Antonyan, D. H. Chang G. Krishnan, J. Sun, J. Hazra, X. Du, M. Liehr, Z. Li D. Kleyko, E. Osipov, A. Senior, A. I. Khan, and Y. A. Sekercioglu, “Y. Kim, M. Imani, N. Moshiri, and T. Rosing, “Geniehd: Efficient data pattern matching accelerator using hyperdimensional computing,” in 2020 Design, Automation Test in Europe Conference Exhibition (DATE), 2020, pp. 115–120.

[21] D. Kleyko, E. Osipov, A. Senior, A. I. Khan, and Y. A. Sekercioglu, “Holographic graph neuron: A bioinspired architecture for pattern processing,” *IEEE Transactions on Neural Networks and Learning Systems*, vol. 28, no. 6, pp. 1250–1262, 2017.

[22] P. Koehn, “Europarl: A Parallel Corpus for Statistical Machine Translation,” in *Proceedings of the Tenth Machine Translation Summit*, AAMT. Phuket, Thailand: AAMT, 2005, pp. 79–86. [Online]. Available: http://mt-archive.inf.tu-darmstadt.de/MTS-2005-Koehn.pdf

[23] G. Krishnan, J. Sun, J. Hazra, X. Du, M. Liehr, Z. Li et al., “Robust rram-based in-memory computing in light of model stability,” in 2021 IEEE International Reliability Physics Symposium (IRPS), 2021, pp. 1–5.

[24] Y. LeCun and C. Cortes, “MNIST handwritten digit database,” 2010. [Online]. Available: http://yann.lecun.com/exdb/mnist/

[25] K. Lee, J. H. Bak, Y. J. Kim, C. K. Kim, A. Antonyan, D. H. Chang et al., “18t high density embedded sram in 28nm ts16 technology,” in 2019 IEEE International Electron Devices Meeting (IEDM), Dec 2019, pp. 2.2.1–2.2.4.

[26] S. Lobov, N. Krilova, I. Kastalskiy, V. Kazantsev, and V. A. Makarov, “Latent Factors Limiting the Performance of sEMG-Interfaces,” vol. 18, no. 4, pp. 1122.

[27] A. X. Manabat, C. R. Marcelo, A. L. Quinquito, and A. Alvarez, “Performance analysis of hyperdimensional computing for character recognition,” in 2019 International Symposium on Multimedia and Communication Technology (ISMAC), 2019, pp. 1–5.

[28] T. Mikolajick, S. Slesazeck, M. H. Park, and U. Schroeder, “Ferroelectric hafnium oxide for ferroelectric random-access memories and ferroelectric field-effect transistors,” *MRS Bulletin*, vol. 43, no. 5, p. 340–343, 2018.

[29] S. Mueller, “Ferroelectric hfo2 and its impact on the memory landscape,” in 2018 IEEE International Memory Workshop (IMW), May 2018, pp. 1–4.

[30] S. Natarajan, M. Agostinelli, S. Akbar, M. Bost, A. Bowonder, V. Chikarmane et al., “A 14nm logic technology featuring 2nd-generation finfet, air-gapped interconnects, self-aligned double patterning and a 0.0388 um2 sram cell size,” in 2014 IEEE International Electron Devices Meeting, 2014, pp. 3.7.1–3.7.3.

[31] K. Ni, A. Gupta, O. Prakash, S. Thomann, X. S. Hu, and H. Amrouch, “Impact of intrinsic variation sources on the device-to-device variation in ferroelectric fet,” in 2020 IEEE International Reliability Physics Symposium (IRPS), 2020, pp. 1–1.

[32] X. Ni, Y. Yin, A. F. Laguna, S. Joshi, S. Dünkel, M. Trentsch et al., “Ferroelectric ternary content-addressable memory for one-shot learning,” *Nature Electronics*, vol. 2, no. 11, pp. 521–529, 2019.

[33] U. Quasthoff, M. Richter, and C. Bie mann, “Corpus portal for search in monolingual corpora.” in *LREC*, 2006, pp. 1799–1802.

[34] A. Rahimi, S. Benatti, P. Kanerva, L. Benini, and J. M. Rabaey, “Hypermensional biosignal processing: A case study for EMG-based hand gesture recognition,” in 2016 IEEE International Conference on Rebooting Computing (ICRC), Oct. 2016, pp. 1–8.

[35] D. Reis, K. Ni, W. Chakraborty, X. Yin, M. Trentsch, S. Dün kel et al., “Design and analysis of an ultra-dense, low-leakage and A. Rahimi, P. Kanerva, and J. M. Rabaey, “A Robust and Energy-Efficient Classifier Using Brain-Inspired Hyperdimensional Computing,” in *Proceedings of the 2016 International Symposium on Low Power Electronics and Design - ISLPED ’16*. San Francisco Airport, CA, USA: ACM Press, 2016, pp. 64–69. [Online]. Available: http://dl.acm.org/citation.cfm?doid=2934583.2934624

[36] Simon Thomann is currently working towards his Master’s degree in Computer Science at Karlsruhe Institute of Technology (KIT). Germany. He earned his Bachelor of Science in CS at KIT in 2019. Alongside his studies, he is working as a student assistant in the Chair of Semiconductor Test and Reliability (STAR) at the University of Stuttgart. His research interests are circuit design, emerging technologies, and cross-layer reliability modeling from device to circuit level.

ORCID 0000-0002-5649-3102

[37] Paul R. Gessner (M’21) received the Dipl. Inf (M.Sc.) degree in computer science in 2017 at TU Dresden, Germany. In 2018 he started his PhD research at the Chair for Embedded Systems (CES) at Karlsruhe Institute of Technology, Germany. Since 2020 he continues his PhD at the Semiconductor Test and Reliability (STAR) chair within the Computer Science, Electrical Engineering Faculty at the University of Stuttgart. His research interests include emerging technologies, system architecture, and intelligent IC test methods. He is a member of the IEEE. ORCID 0000-0002-7175-7284

[38] Hussam Amrouch (S’11–M’15) is a Junior Professor heading the Chair of Semiconductor Test and Reliability (STAR) within the Computer Science, Electrical Engineering Faculty at the University of Stuttgart. He received his Ph.D. degree with distinction (Summa cum laude) from the Karlsruhe Institute of Technology in 2015. He holds seven HiPEAC Paper Awards and three best paper nominations at top EDA conferences: DAC’16, DATE’17 and DATE’17 for his work on reliability. He has 145+ publications in multidisciplinary research areas, starting from semiconductor physics to circuit design all the way up to CAD and computer architecture. Jun.-Prof. Amrouch has delivered 9 tutorial talks in major EDA conferences like DAC and DATE and 25 invited talks (including 2 Keynotes) in several international conferences, universities, and companies. He is a member of the IEEE. ORCID 0000-0002-5649-3102.