Research Narrative

1 Background and Motivation

1.1 Overview

This collaborative project between KAUST and UC Irvine aims at fundamental advances targeting efficient, in-memory, Machine Learning (ML) accelerators to enable edge and across-network *intelligent* computation. Over the past three decades wireless and networking technologies have experienced exponential growth, fueled by breakthroughs in semiconductors, networking and communications theory. Moving beyond connectivity, new generations of networks promise to reinvent entire industry sectors, by creating novel use cases such as massive internet of things (IoT) connectivity, augmented reality, virtual reality (AR/VR), autonomous driving and Vehicular to anything (V2X), etc. [1–4]. These new scenarios demand network support and computational resources that can span orders of magnitude across several dimensions. For example, while AR/VR and autonomous driving applications require voracious bandwidth and extremely low latency processing, IoT systems demand relatively lower bandwidth, at high levels of reliability and efficiency. These use cases promote the rise of applications where sources transmit streams of data through a network with a focus on feedback based, continuous processing, as opposed to traditional applications that involve large scale storage and delayed processing. The distributed nature of these applications present a challenge to existing computational approaches that only consider the cloud and the very edge of the network.

Traditionally, cloud computing has been used to address the computational aspects of the problem by offering highly scalable, low cost computing resources, albeit at the cost of latency, not necessarily in the computing itself, but rather in transferring the data to the cloud and back. These limitations led to the adoption of edge computing as a means of offering services as close as possible to where and when they are needed [5–8]. This could mean at the source, such as on a sensor node with additional processing resources [1,2]. The definition could be expanded to include performing processing across various layers of the network infrastructure, both horizontally and vertically, such as in smart gateways [9], or in network switches or routers, or in 'cloudlets', which are dedicated computing server resources placed a few hops away from the data sources. These resources can vary in scale from a single box placed on a factory floor to a small scale datacenter comprising multiple networked machines. Edge and across-network services reduce data bandwidth needs, improve latency and ease cloud computing requirements, becoming an important pillar in state of the art networks.

Simultaneous with the progress on networks, advances in Artificial Intelligence and Machine learning (AI/ML) techniques resulted in architectures that are exceptionally good at solving complex problems. For current and future network applications, not only is the computational load large (e.g. AR/VR), it is also time varying (e.g. wireless and mobile applications) and more often than not, involves objective functions that are non-linear and/or polynomial, requiring complicated ML heuristics. Thus, interest in *distributed* ML has accelerated to support the naturally growing geographical distribution and sheer size of training data, the need to provide more mobility, security and privacy, lower cost of equipment, energy awareness, resilience against processing and/or communication link failure (no single point of failure) and increase in aggregate I/O bandwidth. In contrast to centralized systems that use a hierarchical aggregation structure, decentralized system allow for aggregation across the network.

Based on the preceding discussion, and as we will argue in the remainder of the proposal, we believe that **Across Network ML (ANML)**, offers a scalable solution supported by distributed ML learning and inference that will fuel the next phase of innovation, where both the network and infrastructure become more scalable, intelligent, distributed and energy efficient. However, ML accelerators comes with their own set of challenges, including training times, dynamic workload, node communication overhead, high computational power requirements (affecting edge nodes), computing scalability and reconfigurability, among other issues. *Thus, there exists a need for a re-configurable and scalable ANML computing architecture that can be deployed across network devices and strike the right balance between latency and accuracy depending on the application needs.* ***Developing such a system lies at the core of our proposed work.***

1.2 Prior and Related Work

Progress in artificial intelligence has been driven by novel machine learning techniques enabled by advances in computing and hardware acceleration. However, as the models become more complicated, the parameter design space grows exponentially, demanding a more distributed means of managing complexity, based on parallizing the training process as opposed to a cloud central approach. At a fundamental level most ML techniques perform linear algebra based data manipulation and transformations on vectors, matrices and tensors, which has been actively researched for decades resulting in specialised hardware accelerators such as Graphical Processing Units (GPUs) and Tensor Processing Units (TPUs) etc. GPUs excel at hardware acceleration via parallel threads. Over the years, GPUs have evolved from highly efficient SIMD (Single Instruction Multiple Data) architectures to more general purpose systems that support reduced overhead branch divergence, such as the Nvidia Tesla GPU series that is designed as a hardware accelerator targeting parallel computing [10].

Due to the heavy matrix manipulations involved in machine learning, Application Specific Integrated Circuits (ASICs) have an inherent advantage over general purpose approaches. This is evident in numerous recent ASIC based approaches including TPU's, that use a Multiple Instruction Multiple Data (MIMD) architecture based on efficient Matrix Multiply units organized as systolic arrays to accelerate ML calculations. The MIMD architecture allows for efficient diverging branch support. Using TPUs as accelerators yields an efficient, high performance computing solution. For example [11] shows a 200x improvement in performance per watt as compared to traditional CPU based systems.

To further improve performance, a number of digital heterogeneous multi-core architectures demonstrated their potential in efficient acceleration and training of large-scale neural networks. In particular, DaDianNao overcame the memory wall problem of general-purpose computers and provided a speedup of 450.65x and energy efficiency of 150.31x [12] compared to a GPU running a large neural network layer. Its architecture is comprised of 16 interconnected tiles, each including eDRAM buffers for storing neurons and synapses and a Neural Functional Unit (NFU) for parallel arithmetic computing. This configuration inspired Resistive RAM (ReRAM)-based multi-core neural network accelerators. Unlike digital counterparts, ReRAM crossbar arrays allow integration of synaptic weight storage and matrix-vector multiplication (MVM) in ReRAM crossbar arrays. Therefore, ReRAM based accelerators could provide more energy and area efficiency. Moreover, the In-memory computing paradigm addresses the memory-wall bottleneck offering orders of magnitude higher operations per seconds by performing all computations locally in memory. ISAAC is one of the first such successful ReRAM-based accelerators. It outperformed fully digital DaDianNao with improvements of 14.8x, 5.5x, and 7.5x in throughput, energy, and computational efficiency, respectively [13]. Around the same time, the PRIME architecture was introduced [14]. Its performance depends on the workload and decreases with increase of the Neural Network (NN) size due to the cost of the data communication between banks/chips. PRIME and ISAAC seem to have similar architecture because both of them took advantage of resistive crossbar arrays (RCAs) for fast and efficient dot-product operation. However, they use different encoding techniques, peripheral and communication interfaces and pipeline structure. In particular, output of RCAs in ISAAC was sensed by Analog to Digital Converters (ADCs), whereas PRIME used Sense Amplifiers (SAs).

The majority of subsequent inference-only accelerators such as AEPE [15] and CASCADE [16] aimed to improve limitations of the design of ISAAC. For instance, AEPE (*diff*=2) achieved the power efficiency of 2.71x and the area efficiency of 2.41x by decreasing the number of Digital to Analog Converters (DACs), lower resolution ADCs and different communication links [15]. CASCADE architecture considered all advantages and disadvantages of ISAAC and PRIME architectures and proposed utilization of Trans Impedance Amplifier (TIA) interface. This helped to reduce energy consumption by 77.5× compared to ADC interface and by 325.4× compared to SA interface. PUMA/Panther is a spatial processor and provides more flexibility and scalability to accelerate a wide range of workloads and different type of data [17]. All these architectures were designed to accelerate only the inference phase of neural networks.

Other hardware architectures, such as PipeLayer [18], AtomLayer [19] and Panther [20], were proposed to support both inference and training phases. PipeLayer achieves 42.45x speedup and 7.17x average energy saving when compared with GPU platforms. AtomLayer also supported efficient inference and training due to a special row-disjoint kernel mapping scheme and data reuse system [19]. AtomLayer provides 1.1× and 1.6× higher power efficiency in inference and training modes compared to ISAAC and PipeLayer, respectively. The most recent architecture, PANTHER, proposed the use of a bit-slicing Outer Product Accumulate (OPA) operations to achieve higher precision and to implement different training algorithms [20]. The proposed training scheme with inter-layer parallelism and weight reuse system was evaluated on PUMA inference accelerator. PANTHER achieved up to 8.02×, 54.21×, and 103× energy reduction and 7.16×, 4.02×, and 16× speed up compared to digital accelerators, ReRAM-based accelerators, and GPUs, accordingly.

A complete summary of these accelerators is depicted in Table 1 showing the hardware models used, the supported workloads, peak performance and throughput. Figure 1 shows the peak computational and power efficiencies for the aforementioned resistive accelerators. These hardware architectures show order of magnitude computational efficiency compared to recent GPU architectures [20]. *However, they lack the on-the-fly re-confgurablity, mixed precision inference and training which are very desirable for energy efficient systems, especially if intended to be deployed across a network where workloads are inherently variable.* In addition, such architectures are evaluated alone without involving host processor, software-stack and other components which is not representative of the case in real-time scenarios.

1.3 Reconfigurable Distributed Machine Learning

Fundamentally, there are two ways to perform distributed ML, either by partitioning the data across different nodes carrying the same model, or partitioning the model across different nodes that have access to the same data. Nodes could connect to each other using different topologies such as tree or ring, or they could be fully decentralized in a peer to peer network, where workers can communicate directly with each other, albeit at a cost of communication. A hybrid option is to layer the model across different nodes where each node performs a sequential part of the needed layers. An example of a layered approach is split learning [21], where a topology is chosen such that the layers of the neural network are spread across two (or more) nodes. In its simplest form (assume a node and a cloud), the training phase involves forward propagation of the data through layers on the node, then transferring an output tensor and associated label to the cloud to continue processing. Gradients are computed on the cloud and returned to the node via back propagation. The choice of where the labels reside (client or server), how the data can be vertically partitioned across multiple data sources, which layer to cut at etc. are all design choices that require reconfigurability at the processing node. In all approaches the choice of topology affects how nodes are connected together, the level of communication between them and the performance and failure resiliency of the system.

The performance of ML algorithms is highly dependent on a number of choices. For example, in Stochastic Gradient

Descent (SGD), choices of the batch size, learning rate, model initialization etc. are different for each domain, dataset and ML model used. Hyperparameter tuning is a key step and typically uses either the first or second order derivative of the function that maps the parameter value to the algorithm accuracy. Other approaches include coordinate descent, Markov-Chain Monte-Carlo, grid and random parameter search and Bayesian hyperparameter optimization among other. Depending on the application at hand and its dynamic needs, ensemble models that combine different models may be needed. This is especially the case when considering distributed data sources. In this case training occurs first locally where the data sources are, then aggregated globally by applying ensemble methods such as bagging, boosting, bucketing, random forests, stacking and learning classifier systems among other. At the network level, to improve accuracy, one could either increase the training data set, increase the model size or both.

Distributed ML provides a scalable solution to address the issues outlined above as long as the communication time does not dominate the computation time. The higher the degree of synchronization between nodes, the faster they could converge to a solution and the easier it is to reconfigure the network, at the cost of increased communication. To schedule and balance the workload, one needs to a) identify the tasks to be executed in parallel, b) decide on the execution order and c) ensure continued load balance across participating nodes.

The allocation of streaming tasks to networked processing nodes has been explored in a variety of existing work. Applications are represented as a graph of tasks with edges representing dependencies, while networks are represented as a graph of compute nodes with edges representing links. Earlier models focused on load balancing in task placement, primarily for the allocation of tasks to multiple servers in a datacenter environment. However, ignored network costs, making them unsuitable for scenarios that consider larger scale networks where communication is more significant. Other work on network aware placement was tailored towards networks of machines that are more widely distributed, but focuses primarily on optimization of a single metric such as bandwidth or latency. To enable these optimizations to be performed online, they are significantly simplified and consider only homogeneous processor platforms. More generalized placement models using integer linear programming have been proposed but are limited as they assume a fully connected cluster of machines, and their models of computing resources and tasks are coarse grained. The across-network approach we consider allows a coarse grained machine learning task to be decomposed into fine-grained tasks that can then be placed at different levels of the network in a manner that optimises overall application performance, for either training or inference. The variability of the hardware accelerator availability in different nodes (or lack of it at some nodes), as well as the dynamic nature of workloads, requires a new model focused on this class of applications, that understands the computational aspects of the accelerator and the communication constraints of the different levels of the network.

The need for ANML: The preceding arguments both at the network level and the hardware accelerator level further support the need for a reconfigurable ***ANML*** approach, which promotes the co-design of the ML algorithms and the platforms and systems on which they reside. Initially, functionality of ML systems were the main concern, however as functionality improved, naturally the design community has become more aware of the resources needed to achieve a given functionality.

1.4 Proposed Work and Team Composition

The proposed research will develop scientific and technological advances required for the design of ANML based acceleration architectures targeting dynamic load, real-time, latency sensitive applications. The proposed research will go beyond the state-of-the-art to develop a scalable, reconfigurable, ML accelerator framework following a holistic approach that supports diverse machine learning network structures and computational models. Feedback between the hierarchical design thrusts will ensure a holistic design approach as detailed in the bench-marking and validation sections of the proposal. Specific research thrusts are:

- **Thrust 1 (T1):** ***Layered distributed machine learning network modelling:*** Establishing a theoretical understanding of the impact of distributing ML across the network. Developing a modelling tool that captures the tradeoffs for both static and dynamic resource allocation.
- **Thrust 2 (T2):** ***Re-configurable ML Hardware Acceleration:*** Proposing a scalable, re-configurable efficient implementation in massively parallel distributed and hierarchically structured ML hardware to support ANML.
- **Thrust 3 (T3):** ***Design space exploration and validation:***
  - ***Architecture exploration and validation:*** Develop a benchmarking and modelling framework using Gem5 to benchmark the proposed architecture.
  - ***Network validation:*** As a proof-of-concept, efficient implementations of the proposed algorithms will be demonstrated on a field-programmable gate array (FPGA) running an ANML scenario.

Intellectual Merit: The intellectual merit of this research is to establish the foundation for creating reconfigurable ML accelerators that can support real-time, dynamic load, applications through ANML. When fully developed, our system and its underlying elements *will make significant enhancements over existing mechanisms for processing dynamic load, latency sensitive applications, leading to the feasibility of next-generation hardware support services and applications.* We anticipate that widespread use of ANML enabled platforms will spawn a broad range of applications of distributed intelligence.

Team Composition: This proposal bring together a multi-disciplinary team of global experts covering the essential areas of machine learning architectures, wireless, networking and computer architecture. At KAUST it will support a team of 3 PI's, 3 postdocs and 3 students spanning the expertise of wireless mobile computing and communication platforms (PI Eltawil), machine learning systems (PI Salama and Eltawil) and hardware acceleration (PI Fahmy) [22–78]. The KAUST team will collaborate with Professor Kurdahi of the University of California, Irvine who is an expert on computer architecture, whose prior work includes re-configurable processing arrays and fault tolerant computing [28, 30, 35, 56, 59, 79–81]. The PIs have complementary expertise and a long history of joint work in relevant areas to this proposal, with a proven track record of success and high impact publications [28, 30, 35, 37, 54–60, 80–97], summarized as shown in Fig. 2.

Together the team brings decades of experience covering the essential areas of ML, wireless systems, networking, hardware acceleration and computer architecture, which are all needed areas of expertise to ensure success of an endeavor of this scale.

2 Research Plan and Methodology

2.1 Thrust 1 (T1): Layered Distributed Machine Learning Systems Modelling

This thrust revisits modern machine learning systems modelling from the perspective of the more fine grained distributed computing enabled by our in-network accelerator approach. In contrast to the assumptions present in existing work on distributed machine learning, where individual nodes represent fully functional general purpose machines, potentially equipped with accelerators such as GPUs or TPUs, we intend to explore what is possible when the overhead of distribution is significantly reduced, as in the case of our proposed architecture. Most established distributed machine learning approaches focus on data parallelism as they assume homogeneity among compute nodes and the network being used only to communicate updates between nodes during training or for query data during inference. The addition of hardware accelerators can increase per-node performance, but due to the added communication latency within the node, requires increased batching to ameliorate this. Hence the data-parallel approach fits better such a traditional view of computing hardware.

When an accelerator is able to connect directly to the network with minimal overhead, it becomes feasible to split an overall computation across multiple accelerator nodes since the full stack latency is reduced due to the lack of a host server (and requisite software network stack). This enables model layering and parallelism, where parts of a single model model are executed on multiple nodes. While some work on model-parallel distributed machine learning has been explored, this has been restricted by the overhead of traditional networking abstractions that we seek to overcome.

When enabling pervasive intelligence in the network in this manner, we expect to deal with large numbers of data flows requiring inference in parallel or alternatively, multiple streams being used to train a distributed model. This requires us to understand the allocation of tasks to the available accelerator nodes, managing the flows of data between them, and how various forms of parallelism impact machine learning performance. *The reconfigurability of our in-memory architecture is fundamental to enabling multi-user leverage of these distributed resources, and the overhead of this reconfiguration impacts the cost of sharing.*

Furthermore, existing work tends to assume a homogenous collection of computing nodes with a shared equivalent network, whereas the model we seek to develop will allow a mix of node types and the network capabilities can vary. Hence we will require a revisiting of the systems assumptions in machine learning to demonstrate what is feasible in the proposed approach, and how varying network and hardware characteristics can best be exploited to provide guarantees about application performance.

2.1.1 Methodology

Scenarios that we intend to consider will be compromised of a set of distributed data generating sources producing continuous streams of data, connected to a network that has a number of intermediate nodes as shown in Figure 3. Nodes may be regular processing nodes or ANML enabled nodes that can be reconfigured to perform different machine learning tasks including both training and inference. Nodes may be configured in data parallel, model parallel or split (model layered) configurations. Depending on the type of ML algorithm being used, each configuration will have implications on how data is partitioned and streamed, how the model is partitioned, where labels are stored, and how synchronization and model freshness is preserved between nodes.

As tasks are processed, individual tasks affect the data volume through a reduction factor that determines the ratio of input data to output data, which reflects the properties of many distributed stream processing tasks. In order to evaluate alternative allocations of resources and tasks, we will consider key metrics of interest, including latency, bandwidth, energy and other considerations such as the financial cost of the computation and the priority etc. Clearly, the metrics are expected to be refined as we proceed with the research.

For each proposed scenario, we will evaluate different network topologies based on task/operator graph, and hardware platform capability. Metrics (latency, bandwidth, etc.) will be calculated based on the distribution of tasks and hardware platforms assuming a predetermined network topology. The logical topology of the network will be represented as a graph, $G_N = (N, E_N)$, where N is the set of network nodes, with bidirectional communication across the set of edges between them, E_N. Application data travels through these nodes and edges towards a central sink. A node can represent either a single machine in the network, such as a gateway, switch or, server, or a 'tier' or 'level' of the network infrastructure. In this case a node may represent multiple machines but the connectivity between them is not modelled at the higher level in the graph. Using this representation allows the network topology to be represented in a tree structure as suits the application models considered.

To represent the application, we will use a constrained graph to define the partitioned model. In model-parallel representations including split-learning, a node consists of one or more layers allocated to a single node. In data parallel representations, a node can be duplicated to represent the same model (or part model) implemented on multiple nodes. Unidirectional edges represent the flow of input data, or activations between model parts. For training, backpropagation represents a reverse dependency, and hence makes an edge bidirectional as gradients are fed back. Each graph node can be assigned to a network node through an *implementation*, that represents a specific allocation of resources on the IMNA accelerator. These implementations are treated as black boxes that take inputs and produce outputs, and have already been benchmarked to determine an estimate of processing time and energy consumption as in Task 3. We start with metrics based on existing modeling of machine learning workloads, then adapt these to the specific metrics of the IMNA architecture.

The proposed model will be used to study the performance of the network under varying configurations and illustrate the impact of various assumption on the hardware on the overall network metrics. This approach tightly couples Thrust 1 with Thrusts 2 and 3 in the proposal, where Thrust 2, provides insight in the hardware capabilities and Thrust 3 on the architecture and network benchmarks, thus closing the loop.

Thrust 1 Tasks

Under Thrust 1 we will carry out the following tasks:

- **T1.1: System modelling of distributed machine learning:** Established theoretical approaches to distributed machine learning, including data, model, pipeline parallelism as well as split learning will be modelled from the system perspective to include computation and communication aspects, including partially shared models;
- **T1.2: Model partitioning and allocation to distributed resources:** Devise a method that given a neural network and a network of computing components, can determine mappings that meet latency and bandwidth constraints, with consideration for impact on model accuracy.
- **T1.3: Dynamic resource allocation:** Extend the allocation approach to deal with multiple dynamic model workloads overlaid onto a network of accelerator resources, including partial sharing.

Thrust 1 Deliverables

- **D1.1: Technical report and publications demonstrating the impact of varying decompositions on model inference and training for a variety of real world models, (tightly coupled to thrusts 2 and 3)**
- **D1.2: Abstract static mapping tool for networked resources.**
- **D1.3: Mapping tool that takes traces of neural network demand and definition of an ANML enabled network to determine a feasible mapping that minimises overheads.**

2.2 Thrust 2 (T2): Hardware Architecture

This thrust presents the core workhorse of the ANML concept, which is a ML accelerator that is re-configurable based on the workload nature and many other parameters discussed in Thrust 1. The objective is to liberate users from having to select and/or dimension a class of ML algorithms and the underlying neural network architecture a-priori. Work under this thrust aims to design a reconfigurable, ML computing platform for efficient learning and inference, that will lead to a system tailored to real-time dynamic load applications. *Creating a truly flexible architecture that can support a wide range of applications requires a convergence of critical factors in the design and engineering of ML systems: energy efficiency, scalability, and functional flexibility.* Benchmarking will be performed using GEM5 (as detailed later in this Thrust and in Thrust 3) to reflect real-time scenarios and software-overhead. The initial architecture presented here is based on numerous brain storming sessions by the team. Over the course of the project, the proposed architecture will be refined and perfected.

2.2.1 In-memory Neural Accelerator (IMNA)

When analyzing deep learning algorithms, three common computing units are required: 1) deep neural network computing unit that is configurable to support different types of DL such as fully-connected NN, CNN, LSTM, Conv-LSTM, RNNs or NLP for a policy or value estimation and prediction, 2) Loss function computing unit and 3) Optimizer for gradient calculations. Thus, these three building units require: 1) a NN accelerator that can support online training with local memory to avoid the memory-wall bottleneck especially when a large amount of data needs to be transferred for inference and training, 2) a processing unit to perform calculations such as loss function and gradients, and 3) a shared storage memory.

The architecture shown in Fig. 4 presents an overview of the proposed In Memory Neural Accelerator Architecture (IMNA). IMNA is responsible for performing all computations related to neural networks. Furthermore, it supports forward and backward operations for inference and training with high reconfigurability to perform different training algorithms, including conventional backpropgation, Contrastive learning, local learning and feedback alignment [98–101]. The accelerator consists of a hierarchically structured assembly of In-Memory Computing Cores (IMCCs), each IMCC consisting of an assembly of vector-matrix multiplication engines which we refer to as an In-Memory Product Engines (IMPEs), where the interconnects are served by In-Memory Associative Processors (IMAPs), with additional hyper transport links and network-on-chip (NoC) routers at each level in the hierarchy.

In-Memory Product Engine (IMPE) is the main block in the neural network acceleration where the matrix multiplication is performed. The IMPE, shown at the bottom of Fig. 4, consists mainly of a vector-matrix multiplication (VMM) engine which can be realized by either one of the following: 1) analog realizations such as memristive crossbar array[1] that inherently performs VMM by applying an input voltage to the crossbar array rows and sensing the output current in one step [102], as an area and energy efficient, high-bandwidth PIM alternative to digital static random access memories (SRAMs) [103, 104] obviating their restriction to time-multiplexed column-serial, row-parallel access. A summary table showing the most recent fabricated memristive macros is depicted in 2. 2) Analog SRAM-based realization where regular or modified SRAM cells are used to store the weights and the input voltages are applied to read the entire columns at the same time, which has the similar concept of memrsitive crossbar arrays expect that the input data has to be binary unlike the memristive arrays can run with analog data as well. And 3) all digital realization, where systolic multiplier and adders units are used to perform VMM operation. Table 3 summarizes the most recent fabricated SRAM-based marcos for DL/AL application with both analog and digital realizations. The energy efficiency of both analog and digital marcos shows orders of magnitude higher than the current GPUs and TPUs. In addition, such analog hardware implementations offer the opportunity to realize mixed precision starting from 1 bit up to 16bits which can be configured on the fly as needed by applications.

Crucially, the analog crossbar VMM implements both forward-propagation inference and back-propagation learning by simply reverting the direction of inputs and outputs to the array [105, 106]. The output current of the analog VMM is then converted to compact digital form to be communicated to added to other VMMs' outputs. New ADC architectures, such as time-based ADCs [107–109], offer low power and compact alternatives for regular ADCs such as Flash and SAR ADCs which are power and area hungry. The conversion in Time-based ADCs is performed on two steps; voltage-to-time conversion, VTC, which simply works as either a pulse width modulator or pulse position modulator, followed by a time-to-digital conversion, TDC which converts the serial binary data to binary-coded data. In our design, we plan to split the T-ADC to have a dedicated VTC circuit for each column in the analog IMPE and to share the TDC circuit among the IMPEs before sending the data to the AP. This would help to have the highest throughput possible, which is usually achieved through the analog communications as discussed in [16, 110], without sacrificing the signal quality at expanse of extra small amount of power consumption. VTC power consumption is in range of 200mW [107, 109] which same as using trans-impedance amplifier to communicate with an analogy signals as discussed in [16]. In addition, sharing the TDCs among different IMPEs will further reduce the overall power and does not require larger bus. Such ADC architecture would enable the needed re-configurability in our architecture. Peripheral programming circuits (PPCs) are used for programming and updating the memory cells (i.e resistive devices for crossbar array or SRAMs). It is worth highlighting that IMPEs can be used as memory units without any change in the hierarchy to perform loss computations.

In the forward path (i.e inference), the output vector of j^{th} layer is given by $\boldsymbol{y_j} = f(\boldsymbol{x_i}^T \boldsymbol{G_{ij}})$ where $\boldsymbol{x}$ is the input column vector of i^{th} layer, $f(\cdot)$ is the ADC function and $\boldsymbol{G}$ is the weight matrix which is stored in the resistive device's conductance [2]. In the backward path, two operations are performed; 1) calculating the back-propagated gradient vector $\boldsymbol{g}$ to the preceding layer $\boldsymbol{g}_i = \boldsymbol{G}_{ij}^T \boldsymbol{g}_j$ which can be carried out by using the crossbar array in the reverse mode by applying the gradient to the output ports of the crossbar array and reading the input ports. And, 2)weight update by $\Delta \boldsymbol{G}_i j = \boldsymbol{\delta}_j \boldsymbol{x}_i^T$ where $\boldsymbol{\delta}$ is the local back-propagated error which can be done through in-place vector-vector multiplication in one step relying on the rank one weight update matrix. Different techniques can be implemented to have energy-efficient back-propagation such as ternary error quantization [111], error-triggered learning [46, 112] and random back-propagation [113]. It is worth mentioning that each IMPE is implemented using RRAMs or SRAMs cells which have limited precision (i.e., 2bit as reported in previous resistive architectures (see Table 1)). Thus, multiple IMPEs are used to realize the required precision and the shift and accumulation operation is performed on the IMAP which will be discussed in the next subsection.

In-Memory Computing Core (IMCC), shown in the center of Fig. 4, consists of N IMPEs connected thought a network on chip (NoC), local In-Memory associative processor (L-IMAP) and router (R). The dimension of the VMM operation performed in the IMPE is predefined by the hardware limitations and design. Thus, large neural layers are partitioned into small matrices and distributed over the IMPEs. Inside each IMCC, a mesh NoC is used to communicate between the IMPEs and create different types of connections like feed-forward or recurrent connections.

The router is used to control dataflow between the IMPEs during the forward and backward paths and is implemented with a lookup table using either resistive CAM or SRAMs. An example of the data flow and the routing table is shown in fig. 5. It consists mainly of two address columns; ***Source*** and ***Destination*** and compare & match circuit. In the forward path, the source IMPE address is searched in the ***Source*** column, and the corresponding matched destination address is

[1]the memristive switching devices (i.e RRAMs, STT-RAMs, PCMs) are sandwiched between two metal interconnect structured as a crossbar

[2]In practice, two crossbar arrays are need to have positive and negative weights and the outputs of the crossbar arrays are subtracted $\boldsymbol{y}_j = f(\boldsymbol{x}_i^T(\boldsymbol{G}_{ij}^+ - \boldsymbol{G}_{ij}^-))$, for simplicity we assume one crossbar array to explain concepts

read. On the other hand, in the backward path, the ***Destination*** column is searched to find the matched source address.
In-Memory Associative Processor (IMAP) is envisioned to consist mainly of a content addressable memory, interconnection matrix, and controller as shown in Fig. 4. IMAP offers a processing architecture that achieves low cost, energy-efficient realizations of state-of-the-art processors with high re-configurability where techniques for power savings such as bit-trimming [23] can be deployed on the fly without changing the hardware or the architecture. The main functions of IMAP are 1) to store the input data for backward path calculation where the inputs are needed for weight update; 2) to combine partial contributions from IMPEs to construct networks of larger precision and dimensionality; 3) implement the special functions such as the activation functions. Different activation functions, such as ReLU, sigmoid and tanh, can be stored as a lookup table, as well as, other operations such as max pooling; and 4) calculate local and global loss functions to support configurable learning techniques such as back-propagation and feedback alignment [100] which need global loss calculations, and local learning which requires local loss calculations [98]. In this work, we target a multi-core IMAP for highest degree in parallelism and reconfigurability.
Routing Network The fully scalable architecture presented will offer highly energy-efficient and yet highly functionally flexible implementation of a variety of DL models, supporting systems of arbitrary size and general connectivity. In particular, it offers a direct efficient implementation of learning models in a scalable and expandable manner, in which hierarchical constructs of neural assemblies are connected by a routing network. We will investigate the applications of conventional routing networks including 3D mesh networks for communications between IMPEs and IMCCs to support high reconfigurability [114–116] in addition to the brain-inspired routing networks such as HiAER as described in [114, 115]. Such scalable and reconfigurable hierarchical organization of fine-grain parallel distributed in-memory-computing with distributed and hierarchical temporal representations will offer unprecedented levels of efficiency combined with high functional flexibility in temporal learning and inference. To realize this routing infrastructure, we will investigate the applicability of the IMAP to implement on-chip routers in addition to providing other functionality at the periphery of the IMCC's. Depending on the application demands, such NoC architecture may necessitate multiple hops to transmit data between blocks. Thus, long, dedicated "fastlanes" may also be needed to minimize overall latency as used in many coarse-grain reconfigurable architectures such as [79] developed by co-PI Kurdahi's team. Finally, a HyperTransport Link will be used mainly for high-speed communication with the outside world and other computing hardware such as conventional CPUs and DDRAMs in addition to communicating with other IMNAs.
Architecture Evaluation

In order to evaluate the proposed architecture, we consider Gem5 which is a modular platform for processor microarchitecture, system-level architecture and computer-system architecture research. Gem5 will be used due to its flexibility, versatility and ability to simulate different hardware configurations and workloads. Fig. 6 (a) shows a Gem5 simulator configuration for an IMNA instance with one IMCC consisting of 2x2 IMPEs and (b) entire architecture that can run operating systems.

The components of Gem5 [117] can be easily relocated, parameterized, extended, or replaced to suit the needs of the system designer, which means that it is possible to freely configure how many IMPEs should be configured in an IMCC, how the AP is connected, and whether the contents of the main memory can be efficiently fetched. Gem5 supports system-wide power model and thermal model, trade-off analysis from application to system-wide is possible. Well-integrated extensions like McPAT [118] and Ruby memory model [117] allow for fine-grained power and thermal analysis.

The main goal of using Gem5 is to optimize the proposed architecture and study trade-offs between the number of DACs and ADCs per IMPEs, the size of the crossbar, and the number of IMPEs per IMCC in addition to study and benchmark different network on chips that connect the IMPEs and the IMCCs. The optimization will be performed for different working scenarios such as low power scenario, highest throughput and thermal density which is a major concern in such accelerators.

Thrust 2 Tasks

Under Thrust 2 we will carry out the following tasks:

- **T2.1: Gem5 simulation Models:** Developing accurate theoretical macro-models for the IMPE and IMAP which allows the evaluation of performance, power, and energy.
- **T2.1: IMNA Implementation:** Implementation and evaluation different configurations to accommodate different working scenarios including maximizing the computational efficiency or power efficiency.
- **T2.3: NOC exploration and Implementation:** Investigate different NOC and the suitablity for our architcure to maximize the perfomance under different running scenarios. In addition to investigate the use of associative processor for NoC routing and make the necessary modifications to the architecture and refining our associative processor compiler according to the modifications of the processor.
- **T2.4: IMNA Performance evaluation**: Evaluation of the overall architecture with GEM5 under different configurations and compare with running on dedicated hardware. Developing a simulator for the proposed architecture and estimate its figures of merit, such as GOPS/W and GOPS/W/mm^2.

Thrust 2 Deliverable:

Full IMNA simulator with different configurations and operating modes including power, area and delay models for each component of the accelerator. Outputs from this thrust will be fed back in Thrust 1 to inform overall system modelling and performance estimation.

2.3 Thrust 3(T3): Benchmarking and ANML Validation

The Gem5 simulator can be used at the Transaction-Level Modeling (TLM) level, the cycle-accurate level, or a mix of them as needed. Therefore, the system modeling stage progress can provide early stage guidance on design choices. The Gem5-based architecture supports execution of various Instruction Set Architectures ISAs) such as ARM, RISC-V, and x64, well-known subsystems, and operating systems and applications that match, enabling more accurate and comprehensive system-level architecture configuration and analysis with existing framework such as TensorFlow [119] and PyTorch [120], which are popular frameworks in machine learning, on top of our architecture. Fig. 7 shows an overall system architecture including the host processor and IMNA.

As initial work, we have already started a core implementation of Gem5 and our extension written in C++, where the steps to construct the architecture are done by python code. Our parameterizable extensions only require modification of the python code, which means that a large number of simulations can be run in a closed-loop, allowing us to investigate the multi-dimensional design space.

2.3.1 Application Mapping Support

At the top level, the proposed architecture will be visible to the programmer in a fire-and-forget model, where the main processor can initiate computations through Python-callable functions. In order to facilitate the mapping of applications onto the proposed architecture, we will develop a suite of tools that will assist designers in translating high level algorithmic descriptions onto low level commands for the hardware. First, we will identify the mapping primitives for basic ML building blocks. To give users maximum efficiency while also maintaining flexibility and ease of implementation, these primitives will be at different levels of hierarchy so that fast mappings can initially be realized using coarse grain functions (e.g.fully connected NN) but later users can carefully optimize their mappings to optimize delay, energy or throughput using finer grain constructs (e.g. ReLU). Additional flexibility will be introduced by enabling users to program the associative processors, configuring routing as well as edge computations across the hierarchy. To do so, we will leverage our existing associative processor compiler [56], augmenting it with additional instruction sets and library functions to implement routing functions.

It is important to point out here that Gem5 is not only a design time tool, but will also be used post deployment. Specifically, as part of a proposed toolset to map applications onto the proposed architecture (Fig. 8). Starting with an inference or training application and some constraints or target quality metrics (i.e. performance, throughput, power/energy, and/or accuracy), the tool will generate an initial partitioning of major subtasks onto different IMCC's. Those subtasks are themselves then partitioned to the different IMPEs and IMAPs inside each IMCC using a combination of templates (e.g. large VMM using multiple IMPEs, implementation of convolution operations with generalized vector matrix multiplication using single or multiple IMPEs, Output functions, etc..). Finally a set of candidate mappings is generated by varying performance, power and precision constraints. The methods to generate such mappings are based on High Level Synthesis (HLS). These mappings are then fed to the Gem5 simulator to evaluate wrt performance, accuracy, power and energy. If needed, this information is fed back to the partitioning and template mapping to generate new mappings.
Complexity and data scaling: We approach *scalability* of the ML hardware-software developed in this program, from two complementary perspectives. *First, having support for on-line local ML algorithms will ensure continuous and consistent operation on streaming data.* This is in addition to of course supporting traditional ML approaches that require multiple passes through off-line data. Second, our hierarchically structured, sparsely active architecture (detailed in Thrust 2) and its highly efficient hardware realization using cross-bar arrays (modelled in both CMOS silicon and emerging memory technologies) *offers a flexible and expandable physical substrate that readily expands to virtually unlimited network size and connectivity in ML learning and inference.*

Finally, while similar tools have been proposed and developed [121]. There are several features that make this proposed approach different:

1. **Mapping target:** Most tools such as [121] target FPGAs. This tool will map directly onto a real ANN architecture (which may itself be realized as an ASIC or overlayed on FPGA fabric).
2. **Dynamic reconfiguration:** During runtime, the application mapping may be changed for a variety of reasons (e.g. re-training, or changing the tiering of computations, or changing the precision, or changing the input granularity to accommodate different levels of input resolution). Such changes may be due to change in operation conditions or environment such as remaining battery energy or re-tiering of computations due to changing communication link quality. In such cases, new mappings must be reloaded. These new mappings are pre-generated and reloaded, or the current mappings modified (e.g. using more or fewer IMCC's to improve performance or save power). A reconfiguration management middleware layer will be added to manage such reconfiguration, receiving commands from the host processor and applying the changes to the accelerator.

3. **Lifecycle template granularity scaling:** As more designs are run through the mapping tool, a repertoire of the resulting mappings is added to the repertoire of templates. When a new application is input, the partitioning tool's granularity is gradually scaled to look for larger templates matching the application's requirements. For those matches that are not exact, an edit distance between the two subgraph (the exiting template and the application component) can be derived and used to morph the existing template onto the one required by application. A potential side effect of such an approach would be to reduce the mapping time and the number of times the Gem5 simulator needs to be invoked in the loop.
4. **Runtime optimization:** As more applications are run through the accelerator, a meta-learning task is running in the background. This background task may invoke the mapping tool to suggest new mappings or runtime management policies that are more optimized than the ones currently running. Co-PI Kurdahi and his collaborators have been experimenting with such a concept in the Information Processing Factory (IPF) [122] [123] and getting promising results. We propose to explore this concept on the proposed accelerator.

Scenario Benchmarking: The proposed architecture and tools will be validated on a variety of benchmarks in multiple scenarios. As an initial appraocvh, we will build on work done by Co-PI Kurdahi within the IPF project that is finalizing a benchmark suite for autonomous driving which comprises end-to-end autonomous driving tasks. These benchmarks will be particularly useful for both the accelerator benchmarking as well as the network-wide tiering experiments. Using the Sumo driving simulator [124] we will create multiple scenarios such as the ones in Figures 9 and 10 and will validate the reconfiguration requirements under such cases. In Figure 9 the ML tasks are partitioned across the network. Assume that the tasks were initially partitioned in a balanced way across the network (a). If the vehicle's battery is nearly depleted at some point (b), then the tasks must be redistributed so that most of the computation is offloaded to the fog/cloud. Energy saving will result not only from decreasing the number of tasks performed locally, but also from the reduced precision needed for these tasks. If, in another scenario the Communication link to the fog/cloud becomes unreliable, then all the computation is performed in-vehicle (c). There could be other scenarios where the communication link is severed and the car battery is low, in which case the computations must be all carried out at the vehicle but with reduced precision (e.g. with scaled down camera resolution). Figure 10 presents a similar set of scenarios but with a data parallel partitioning where the data is split across the network instead of tasks. What these two sets of scenarios highlight is the need for *agile, reconfigurable ML accelerator that can reconfigure at runtime to scale power, accuracy and performance to deal with the dynamic nature of our cyber-physical world.*

Additionally, we will consider benchmarks from other applications that require widely different network support, specifically medical devices, where reliability, distributed learning and privacy are key, while bandwidth is reduced. In fact, medical applications are ideal to consider for split learning ML where private data from one patient may be split across different data sources (pathology and radiology for example), or different patient information can be used to refine a central ML model. In prior projects [125], we implemented two such devices (at the software and/or compute hardware levels) which we plan to use to illustrate dynamic tiering scenarios: a network-connected pacemaker, and an artificial pancreas. Both pose some very interesting case studies vis-a-vis ML (both local and federated).

The divergent requirements of these two benchmarks will be particularly relevant to highlight the relevance of ANML reconfigurable approaches. It is expected that other benchmarks will be added as the research matures.

FPGA Prototype: We will abstract these sample scenarios to network requirements and test the performance of the proposed system. As a mock scenario, we will setup a test that consists of an edge node that has computational and caching capabilities and an application server. For the application we will consider autonomous driving and medical devices as two exemplars of applications with widely varying requirements as discussed earlier. we will split the data across a large number of generators that send packets to the application server. We will vary the average rate of the traffic to create realistic scenarios. Moreover, to test the hardware response to increased congestion, we will benchmark performance versus load. The increase in congestion is detected by the increase in the packet drop rate, which is reported by the edge node to the controller. In our initial evaluation we focus on the packet loss aspect. Instead of comparing the rate of packet loss of different schemes, we will rather measure the effect of the packet loss rate from the application point of view under the different schemes. We will also consider the effect of the latency of such packets on the application performance metrics. We will evaluate the ability of ML hardware to predict the effect of the lost or corrupted packet on the application performance. We will also evaluate the ML hardware on how it reacts to late packets.

FPGAs further enable us to investigate the distributed aspects of our proposed approach in practice. We will build a network of accelerator nodes comprising suitable network interfaces and the functional model of the in-memory accelerator. We will use this to characterise the communication aspects of different deployments and combine this with the results of the Gem5 simulations of the architecture itself to arrive at meaningful results for the proposed architecture. Here the FPGA serves as both *suitable networking infrastructure and a platform for a real-time functional model, thereby enabling the Gem5 characterisation to be incorporated to reason about the distributed deployment in a realistic scenario.* In previous work, we developed a testbed for measuring FPGA accelerator performance in a networked setting, both as hosted accelerator cards and direct network attached accelerators [126]. We will extend this testbed with the IMNA functional architecture, and expand to multiple nodes. Measurements will be taken with different task flows, measured using an FPGA measuring device that can provide highly precise measurements, enabling near-realtime experimentation.

Thrust 3 Tasks

Under Thrust 3 we will carry out the following tasks:

- **T 3.1: Reconfigurable design space exploration:** Refinement of algorithms and architectures developed in Thrust 1 and 2 towards highly optimized implementation of the proposed IMNA and reconfigure activations, specifically studying how different algorithms such as CNN, FCNN, RNN, LSTM etc. map to the proposed IMNA. Targeting a low power architecture (10-100$\times$ lower power) by utilizing the re-configurable features in the proposed architecture to support an adaptive and dynamic back-propagation. Study impact of energy efficient approaches on overall performance such as bit trimming, model pruning, compression etc. during inference and training.
- **T3.2: Gem5 ANML simulation model:** Integrating Gem5 components into Gem5 ANML simulation platform.
- **T3.3: Develop ANML mapping tools:** Develop first generation ANML mapping tools allowing design time (static) precision tuning.
- **T3.4: Support for dynamic reconfiguration:** Incorporating dynamic reconfiguration to allow precision tuning and partial reallocation of resources at runtime into the ANML support tools.
- **T3.5: Networked FPGA integration:** Integration of the ANML fucntional model with network connectivity infrastructure on FPGA platforms for a complete functional node model.
- **T3.6: Multi-node networked FPGA demonstrations:** Assessment of model allocations from Thrust 1 on multiplue ANML nodes connected via a network to measure throughput, latency, and resource metrics.

Thrust 3 Deliverables

- **D3.1: Gem5 ANML simulator**
- **D3.2: Application to ANML Mapping tools - Version 1 (static mapping)**
- **D3.3: Application to ANML Mapping tools - Version 2 (dynamic mapping)**

3 Significance and Potential Impact

The next phase of innovation is expected to be fueled by concurrent advances in AI/ML empowering network intelligence where both the mobile networks and infrastructure become more scalable, virtualized, intelligent, distributed and energy efficient. ANML offers an opportunity to distribute learning and adaptation across the network, discovering complex correlations and optimizing the end-to-end computational network performance.

Outreach Efforts: The results, tools, hardware and software from this research project will be disseminated through a variety of modalities, including a project's web page, participation in student sessions, Work-in-Progress sessions at leading conferences and workshops, tutorials, special sessions and special issues at major conferences and professional meetings, and articles in leading journals. On a larger theme, through a fusion of both research and educational objectives, this project will simultaneously contribute towards the development of new design methodologies and courses that promote engagement of students at multiple levels.

For student outreach and recruitment, the Principal Investigators (PIs) will conduct a winter school on AI and ML HW implementation similar to a previous event (microelectronics workshop 2020) with 40 students from 12 countries. On a larger scale, the PIs intend to organize a workshop and a summer school in KAUST to both acquire knowledge and spread the main project ideas and output to the KAUST community and beyond. The PIs will build on their experience organizing a a workshop on Biosensors and Bioelectronics in 2018 with more than 100 attendees and 40 international speakers. A separate budget will be requested for such activity as supported by KAUST for workshops and professional gatherings sponsored by KAUST. Finally,during the project, the PIs will integrate results from the project within the Winter Enrichment Period (WEP) to provide ML and AI for all (no engineers) master class. Co-PI Salama organized a similar class in WEP 2020. WEP is the largest outreach event held at KAUST attracting 1 Million social media interactions, 8000 visitors and 400 student registrations in 2020.

To have a truly global impact, we intend to utilize the significant outreach capabilities of our partner university (UCI) to spread awareness of the project and its outcomes. For example, UCI has established routes of outreach via the Undergraduate Research Opportunities Program (UROP), and Mathematics Engineering Science Achievement Program (MESA). Building further, the project PIs and students can reach out to high school students via interactive AI projects, lessons, and presentations. Finally, UCI has a dedicated section for outreach under the division of student affairs that can assist with planned outreach activities.

Technology Transfer and Societal Impact: ML and networking is naturally an area of extreme interest to industry, and the success of this project will help affirm KAUST's position as a hub for national and international know-how in AI/ML. The proposed architecture is general enough to be used in most AI/ML applications due to its extreme efficiency, flexibility and scalability. As a followup to this project, we plan to seek funding to build a full chip and enhance application mapping support. This development will fuel significant research activity not only among the current participants but also in the global research community. Several of the PIs have had prior experience with startup companies in the past, and could catalyze similar endeavors in this case, setting new standards for hardware-based AI/ML and potentially transforming the field.

4 Management Plan

4.1 Overall Project Management

This project is a collaborative effort between KAUST and UCI. The collaborating teams will work closely together based on the project tasks and progress. PI and Co-PIs have clearly defined tasks and milestones, which they are responsible for, as detailed in the Gantt chart. In subsets, the PIs have worked jointly on research efforts, disseminating their results in journals, conferences and workshops. This collaborative track record will be one of the main reasons the proposed project will be successful. As described in Section 1.4, the team brings together synergistic and complementary expertise that is needed to perform the proposed research tasks. As in every research project, there are potential roadblocks that may be encountered during the course of execution. To mitigate these risks we have a number of strategies in place.

- In terms of accelerator and INC verification, we will pursue both GEM5, SystemC and FPGA functional verification. While each will provide unique results, different approaches complement each another by providing detailed feedback and insights on the functionality and performance of the proposed accelerator.
- In terms of application, we will pursue both simulation based verification using ML algorithms to validate algorithm performance, as well as FPGA in the loop emulation to validate system performance.

These strategies taken in conjunction with the fact that the PIs have significant experience in their respective field should minimize risks. Success of the project will be measured by reaching milestones defined in the project plan and task descriptions.

4.2 Collaboration Plan

The collaborative research effort will be led by PI Eltawil who will manage the overall execution of the project. PI Salama will lead Task 1, PIs Fahamy and Eltawil will lead Task 2, PIs Kurdahi and Eltawil will lead Task 3, and PIs Fahamy, Kurdahi and Salama will lead Task 4. This research team has been carefully selected to ensure we have gathered the competences needed for tackling this challenging project. Reaching the ambitious goals detailed in the proposal requires an efficient and smooth collaboration to unlock the synergy potential of the distributed expertise. To ensure synchronized progress and coordination of material and knowledge exchange (supported by student exchange), annual project progress meetings will be held. The annual meeting will result in a project report as well as a detailed set of recommendations for the following year activity. The KAUST teams will meet on a weekly basis. Thrust leaders and their respective teams will be holding meetings every two weeks to synchronize activity. PIs will hold a monthly meeting to report on progress and synchronize work. A full online team meeting for all members, including affiliated students, will be held every quarter featuring talks and selected presentations by team researchers.

4.3 Mentoring Plan

This effort is indeed a global effort. Students and post-docs affiliated with this effort will benefit from an unprecedented opportunity to engage with world experts and their teams leading to exchange, not only of technical information, but also of cultural approaches and work ethics creating a well rounded educational experience. The proposed research will be instrumental in educating students considering the “full picture” across the application, architectural platform and technology levels. The diverse nature of this project, including the integration of chip design, algorithm development, wireless networking, and system validation provides a rich environment for students to develop a strong understanding of research and technology. This will result in scientists who are trained in the “systems” aspects with an in-depth understanding of trade-offs and interactions across layers of a complex system design.

Two postdoctoral researchers and five graduate students will be funded on this project. The goal of the mentoring program will be to provide the skills, knowledge and experience to prepare the student/postdoctoral researcher to excel in his/her career path. To accomplish this goal, the mentoring plan highlighted below is targeted at enhancing the academic experience of junior researchers’ experience, by providing a structured mentoring plan, career planning assistance, and opportunities to learn a number of career skills such as writing grant proposals, teaching students, writing articles for publication and communication skills. To achieve these goals the PIs commit to devoting at least 10% of the time invested in the project towards mentoring the researchers. Specifically, the following goals are set:

- Working with the postdoctoral and student researchers to establish and implement an Individual Development Plan to assess career goals and developmental needs;
- Meetings at least once weekly to discuss progress toward project goals;
- Periodic (quarterly) evaluations by both the researchers and their faculty mentor;
- Providing opportunities to network with visiting scholars who are leaders in our field during relaxed social settings (Lunch, dinner etc.) when they participate in the school’s visiting speaker series;
- Travel to at least one topic related conference or professional meeting (travel funds are included in the budget), with the goal that the postdoctoral and student researchers present a poster or paper at the conference;
- Participation in the PIs’ research group meetings, in which members will be expected to present their research regularly, and feedback and coaching will be given to help all members to develop their communication and presentation skills;
- Students will be directly involved in coordinating and running the activities detailed in the Significance and Impact section, which in addition to the technical know-how gained will lead to organizational and outreach activity skills.

Gantt Chart

Tasks & Activities		Team	Y1				Y2				Y3			
			Q1	Q2	Q3	Q4	Q1	Q2	Q3	Q4	Q1	Q2	Q3	Q4
	Thrust 1: Layered Distributed Machine Learning Systems Modelling (Leads: Eltawil and Fahmy)													
1	System modelling of distributed machine learning	Eltawil Salama	✓	✓	✓									
2	Model partitioning and allocation to distributed resources	Fahmy Eltawil			✓	✓	✓	✓	✓	✓				
3	Dynamic resource allocation	Eltawil Fahmy							✓	✓	✓	✓	✓	✓
	Thrust 2: Hardware Architecture (Leads: Kurdahi and Eltawil)													
1	Gem5 simulation Models	Kurdahi Eltawil	✓											
2	IMNA Implementation	Kurdahi Eltawil	✓	✓										
3	NOC exploration and Implementation	Kurdahi Fahmy			✓	✓								
4	IMNA Performance evaluation	Kurdahi Salama			✓	✓	✓	✓						
	Thrust 3: Benchmarking and ANML Validation (Leads: Kurdahi and Salama)													
1	Reconfigurable design space expration	Kurdahi Salama			✓	✓	✓	✓	✓					
2	Gem5 ANML simulation model	Kurdahi Eltawil			✓	✓	✓	✓	✓					
3	Develop ANML mapping tools	Kurdahi Eltawil					✓	✓	✓	✓	✓			
4	Support for dynamic reconfiguration	Kurdahi Eltawil						✓	✓	✓	✓	✓		
5	Networked FPGA integration	Fahmy Salama								✓	✓	✓	✓	✓
6	Multi-node networked FPGA demonstrations	Fahmy Salama								✓	✓	✓	✓	✓

Table 1: Summary of the ReRAM-based hardware accelerators

Arch. Name	**ISAAC'16 [13]**	**PRIME'16 [14]**	**AEPE'17 [15]**	**PipeLayer'17 [18]**	**AtomLayer'18 [19]**	**Newton'18 [?]**	**CASCADE'19 [16]**	**PUMA'19 [17]/PANTHER'20 [20]**
Workload Types	CNN, MLP, DNN	CNN, MLP	CNN, MLP	CNN, MLP	CNN, MLP	CNN, MLP, DNN	DNN, RNN	MLP,CNN, LSTM, GAN, RBM, RNN, BM, SVM, Linear regression, Logistic regression
Benchmark neural networks	VGG-A/B/C/D, MSRA-A/B/C, DeepFace	LeNet-5, CNN-1, CNN-2, MLP-S/M/L, VGG-D	AlexNet, VGG-16, ResNet50	MNIST-A-B-C-0, AlexNet, VGG-A/B/C/D/E	VGG-19, ResNet-152, DCGAN	VGG-A/B/C/D, MSRA-A/B/C, AlexNet, ResNet34	VGG-A/B/C, MSRA-A/B/C, AlexNet, DeepFace, NeuralTalk	BigLSTM, MLP, NMT, VGG-16, VGG-19, LSTM-2048
Technology	32 nm	65 nm TSMC CMOS	32 nm	32 nm*	32 nm	32nm	65nm	32 nm
Frequency	1.2 GHz	3 GHz	1.2 GHz	1 GHz*	1 GHz*	1.2GHz	n/a	1.0 GHz
Improvement (baseline)	Throughput: 14.8x; Energy: 5.5x; Comp. efficiency: 7.5x (DaDianNao)	Performance ∼2360x; Power efficiency ∼895x (ISAAC)	Power efficiency by 2.71x (ISAAC); Area efficiency by 2.41x (ISAAC)	Comp. efficiency = 3.1 x Power efficiency =0.37 x (ISAAC)	Power efficiency 1.1x (ISAAC); Training efficiency 1.6x (PipeLayer); 15x smaller footprint	a 77% decrease in power, 51% decrease in energy, and 2.2× increase in throughput/area (ISAAC)	consumes 3.5× lower energy (an ADC-based in-ReRAM computation architecture)	up to 8.02×, 54.21×, and 103× energy reductions as well as 7.16×, 4.02×, and 16 × execution time reductions (digital accelerators)
Power, W	65.8	n/a	n/a	82.6	6.89	n/a	n/a	62.5/ 105W
Area, mm2	85.4	n/a	n/a	168.6	4.80	n/a	n/a	90.6/ 117
Inference latency, ms	8.00 (VGG-19) 43.46 (ResNet-152)	n/a	n/a	2.60 (VGG-19) 14.13 (ResNet-152)	6.92 (VGG-19) 4.01(ResNet-152)	n/a	n/a	n/a
Peak perf., GOPs	n/a	n/a	81.92 *256 (AEPE diff =1)	n/a	n/a	n/a	n/a	52310
Power efficiency, GOPs/s/W	627.5 (ISAAC-CE) 312.5 (ISAAC-SE) 644.2(ISAAC-PE)	n/a	1022.47*256 (AEPE diff=1)	142.9	682.5	920	n/a	840
Area efficiency, GOPs/s/mm2	478.9 (ISAAC-CE) 103.35 (ISAAC-SE) 409.67(ISAAC-PE)	n/a	n/a	1485 (PipeLayer)	475.6 (AtomLayer)	680	101	580 (PUMA)
Precision, fixed point	16-bit	16-bit	16-bit	16-bit	16-bit	16-bit	16-bit	16-bit (inference) 16- and 32-bit (training)
RCA	128x128	256x256	128x128	128x128	128x128	128x128	64x64	128x128
RCA cell precision	2-bit	4-bit	3-bit	4-bit	2-bit	2-bit	1-bit	2-bit
Crossbar latency	100 ns	100 ns*	100 ns	100 ns*	100 ns	100 ns*	100 ns*	100 ns*
Input encoding	bit-serial	multi-level	bit-serial	bit-serial	bit-serial	bit-serial	bit-serial	bit-serial
Input precision	16-bit	6-bit	16-bit	16-bit	16-bit	16-bit	16-bit	16-bit
Synaptic weight precision	16-bit	8-bit	16-bit	16-bit	16-bit	16-bit	16-bit	16-bit (inference) 16/32-bit (training)
Output precision	16-bit	6-bit	16-bit	16-bit	16-bit	16-bit	16-bit	16-bit
ReRAM materials, characteristics	TiO/HfO $R_{on}/R_{off} = 2k\Omega/2M\Omega$	Pt/TiO2-x/Pt $R_{on}/R_{off} = 1k\Omega/20k\Omega$ 2V SET/RESET	TiO/HfO $R_{on}/R_{off} = 2k\Omega/2M\Omega$	Pt/TiO2-x/Pt $R_{on}/R_{off} =$ 1kΩ/20kΩ 2V SET/RESET	TiO/HfO $R_{on}/R_{off} =$ $2k\Omega/2M\Omega$	TiO/HfO $R_{on}/R_{off} = 2k\Omega/2M\Omega$	n/a	$R_{on}/R_{off} = 100k\Omega/1M\Omega$ Read Voltage 0.5V
H/W evaluation models and compilers	CACTI 6.5	Synopsis Design Compiler, modified NVSim, CACTI-3DD, CACTI-IO	CACTI, Orion, NVsim	NVsim	CACTI 6.5	CACTI 6.5	Cadence Spectre	PUMAsim,CACTI 6.0 , Booksim 2.0 Orion 3.0
Network-on-chip	concentrated mesh	memory bus	concentrated mesh	n/a	INet, LNet, ONet	concentrated mesh	n/a	2D Mesh
Off-chip links	HyperTransport	HyperTransport*	HyperTransport*	HyperTransport*	HyperTransport*	HyperTransport	HyperTransport*	HyperTransport
Largest network	26 layers (330 million parameters)	VGG-D: 16 layers; 1.4*10ˆ8 synapses ∼1.6*10ˆ10 operations	VGG16 (138 million parameters);	VGG-E (141 million parameters)	VGG-19 (141 million parameters)	VGG-19 (141 million parameters)	VGG-C (138 million parameters)	VGG-19 (141 million parameters)
In-situ training	✗	✗	✗	✓	✓	✗	✗	✓

*Not reported in the publication, but we expected to be that based on the discussion
n/a: Not reported

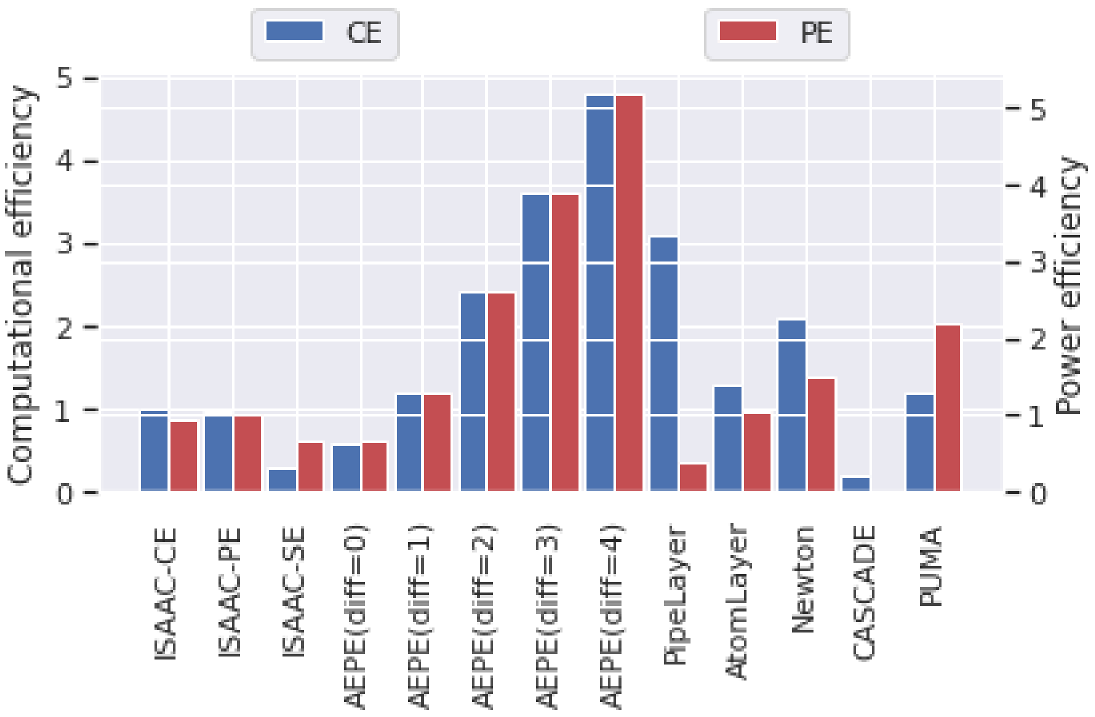


Figure 1: Performance of the ReRAM accelerators: computational efficiency (CE) normalized to ISAAC-CE and power efficiency (PE) normalized to ISAAC-PE.

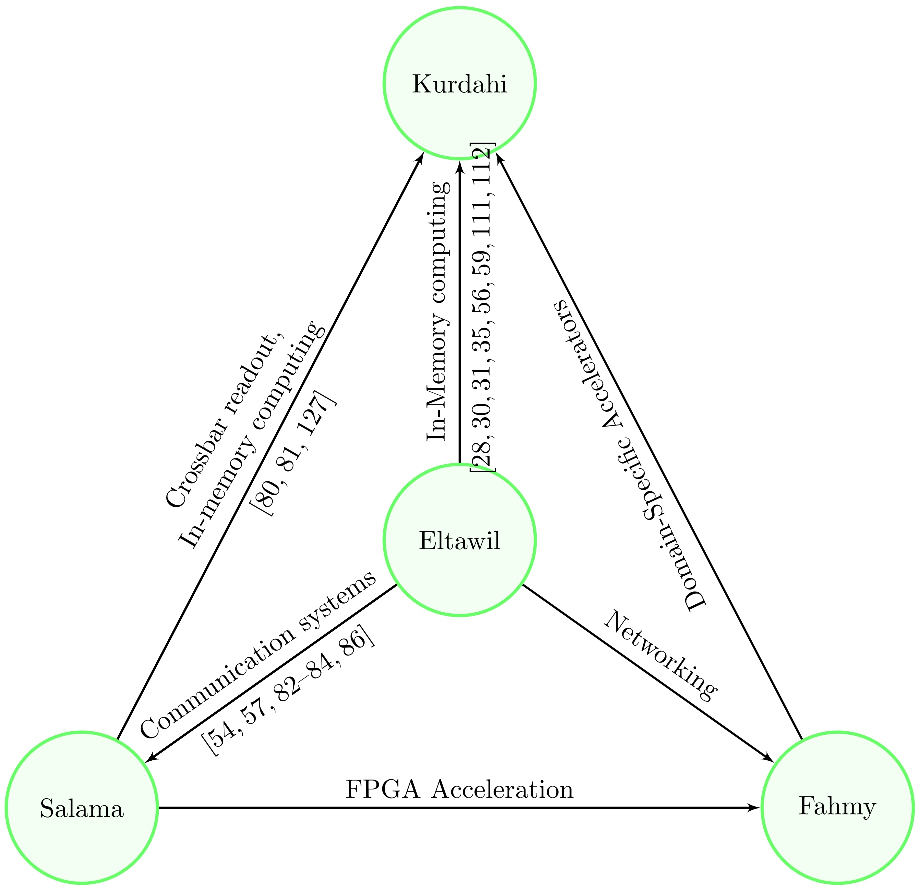

Kurdahi
Eltawil
Salama
Fahmy
Crossbar readout, In-memory computing [80, 81, 127]
In-Memory computing [28, 30, 31, 35, 56, 59, 111, 112]
Domain-Specific Accelerators
Communication systems [54, 57, 82–84, 86]
Networking
FPGA Acceleration

Figure 2: Team composition and collaborations

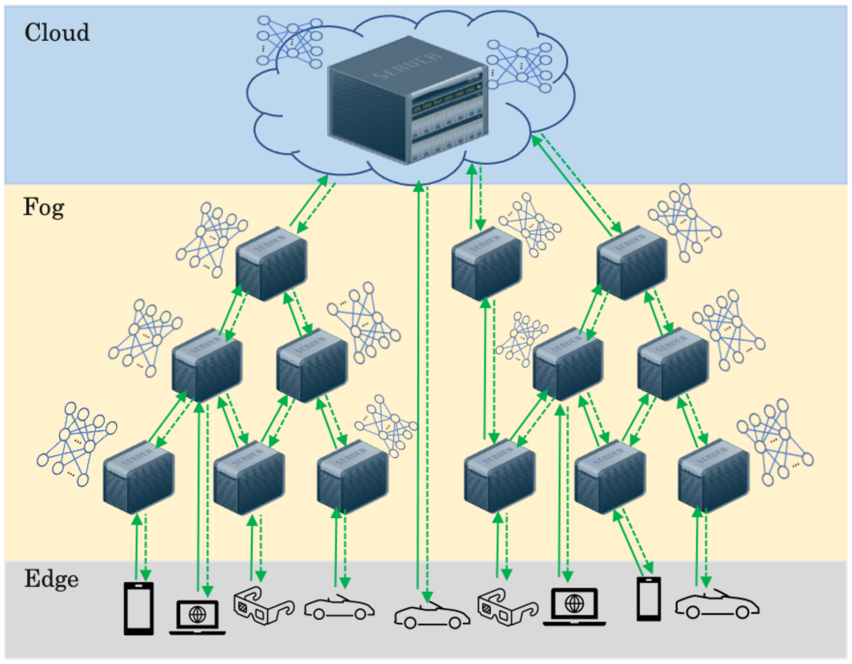


Figure 3: Distributed training and inference across the network for different workloads

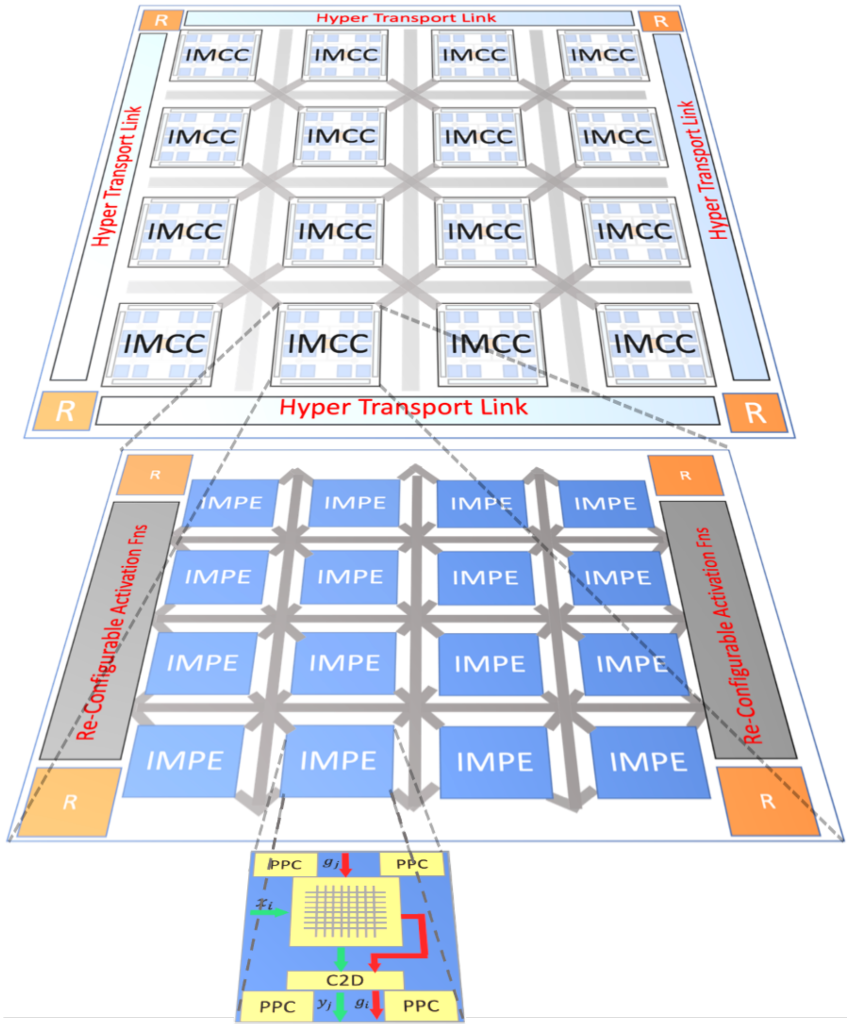


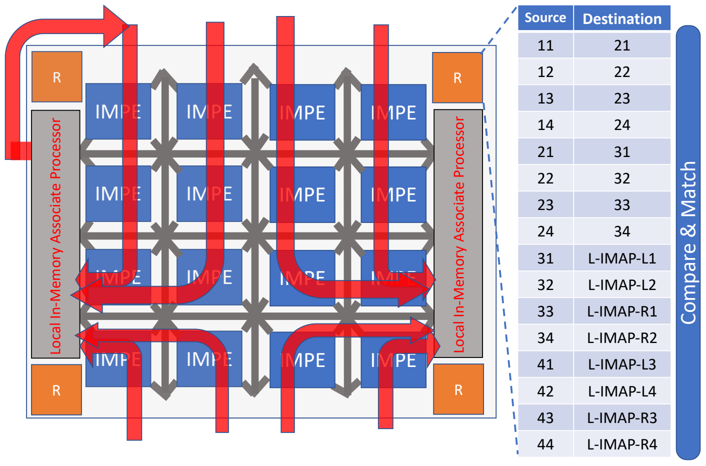
Figure 4: In-memory neural accelerator chip architecture. IMPE: In-memory product engine, PPC: prephrial programming circuits, C2D: current to digital converter, R: router and IMCC: In-memory computing core.

Table 2: Recent Fabricated ReRAM macros for ML/DL Acceleration

	ISSCC'18 [128]	**ISSCC'19** [129]	**ISSCC'20** [130]	**ISSCC'20** [131]	**OJCAS'21** [132]
Technology	65nm	55nm	22nm	130nm	40nm
Capacity	1Mb	1Mb	2Mb	158.8Kb	2Mb
Subarray Size	8x128K	256x512	512x512	N/A	128X128
Cell	1T1R	1T1R	1T1R	2T2R	4T2R
ReRAM size	$0.25\mu m^2$	$0.2025\mu m^2$	N/A	$1.69\mu m^2$	$0.55\mu m^2$
Cell size	N/A	N/A	N/A	$3.38\mu m^2$	N/A
Die area	N/A	N/A	$6mm^2$	$21.82mm^2$	N/A
ReRAM mode	Memory/ CIM	Memory/ CIM	Memory/ CIM	Memory/ CIM	Memory/ CIM
RRAM precision	binary	ternary	4bit	Analog (0.4-4uA)	ternary
ADC	3b	3b	N/A	1b-8b	N/A
Precision (I,W,O)	1,T,3	2,3,3	2,4,10	1,T,8	N/A
Energy efficiency, (TOPS/W)	25.42 @ 1V	21.9 @1V	45.52@0.8V	78.4 @4.2V	223.6 @ 0.7V
Read Delay (ns)	14.8	14.6	13.1	51.1	0.92
Inference Speed	N/A	N/A	N/A	77 μ s/Image	N/A
Accuracy	MNIST: N/A	CIFAR10: 88.52%	CIFAR10: 90.18% CIFAR100: 64.15%	MNIST: 94.4%	MNIST: 95.7% CIFAR10: 81.7%

Table 3: Recent ASIC Macros for ML/DL Acceleration.

	Analog MAC						**Digital MAC**	
	VLSI'16 [133]	**ISSCC'18 [134]**	**ISSCC'18 [135]**	**ISSCC'19 [136]**	**ISSCC'20 [105]**	**ISSCC'20 [106]**	**ESSCIRC'19 [137]**	**ISSCC'21 [138]**
Technology (nm)	130	65	65	55	28	7	65	22
Cell Structure	C6T	S6T	10T	Twin-8T	6T+Transposable cell	8T	6T	6T
Bit-cell size (F^2)	254	125	N/A	548	210	1081	NA	783
Subarray Size	128×128	64×64	256×64	64×60	512×128	64×64	128×128	256×64
Capacity	16Kb	4Kb	16Kb	3.75Kb	64Kb	4Kb	16Kb	64Kb
Precision (I,W,O)	5,1,1	1,1,1	7,1,7	4,5,7	8,8,20	4,4,4	16,16,23	8,16,24
Supported Algorithms	Linear Classify	XNORNN BNN	CNN	CNN	CNN	CNN	MLP CNN	MLP CNN
Energy Efficiency (TOPS/W)	1.42	55.6	28.1	18.4	262.3$\sim$610.5 (4b/4b)	68.44 (4b/4b)	117.3 (1b/1b)	89 (4b/4b)
Throughput (GOPS)	64	1780	10.67	17.6	372.4 (4b/4b)	124.88 (4b/4b)	567 (1b/1b)	3300 (4b/4b)
Accuracy	91% (MNIST)	97.5% (MNIST)	96% (MNIST)	90.42% (CIFAR10)	91.94% (CIFAR10)	98.6% (MNIST)	N/A	N/A
Support	Inference	Inference	Inference	Inference	Training	Training	Inference	Inference


R
R
R
R
IMPE
Local In-Memory Associate Processor
Local In-Memory Associate Processor
Compare & Match

Source	Destination
11	21
12	22
13	23
14	24
21	31
22	32
23	33
24	34
31	L-IMAP-L1
32	L-IMAP-L2
33	L-IMAP-R1
34	L-IMAP-R2
41	L-IMAP-L3
42	L-IMAP-L4
43	L-IMAP-R3
44	L-IMAP-R4

Figure 5: An example of the data flow and the corresponding routing table

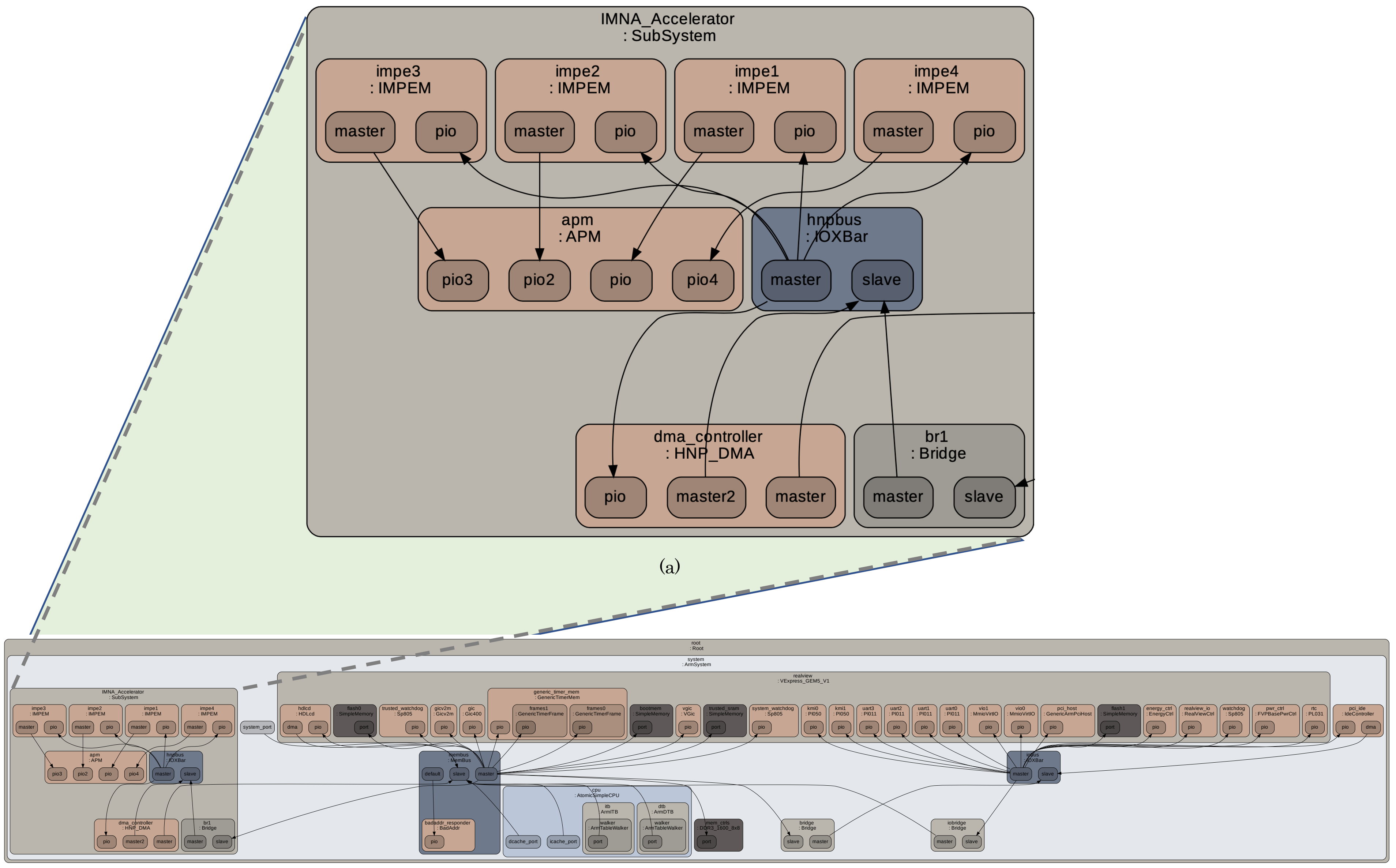


Figure 6: IMNA architecture in gem5: a) initial implementation of IMCC consisting of IMPE, AP, DMA controller, a bus, and bus bridge and b) Real-world system-level gem5 implementation including IMCC, microprocessor, memory controller, real-time clock, timer, interrupt controller, watchdog, PCI host, flash/IDE based storage

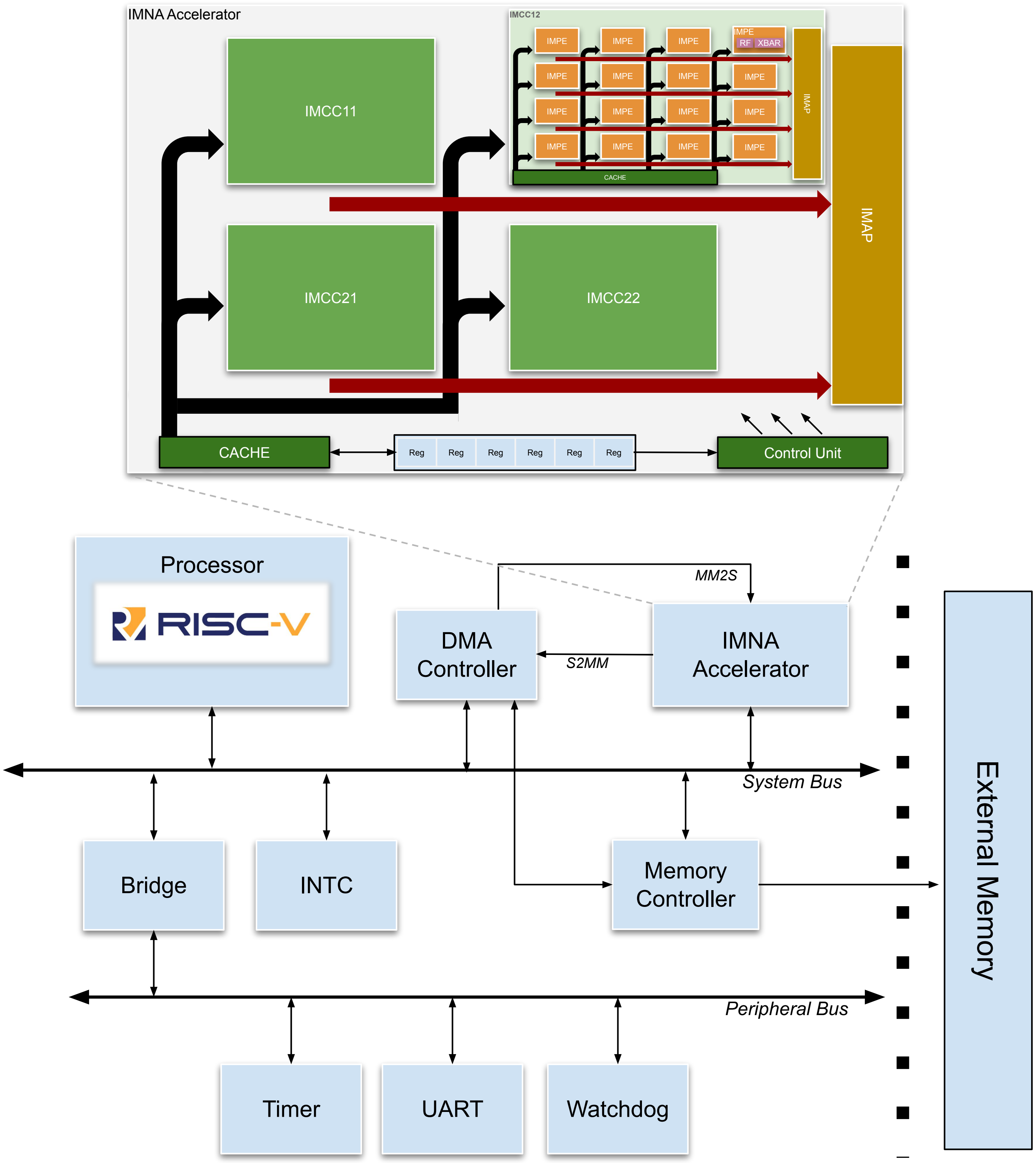


Figure 7: System-level overview of IMNA architecture

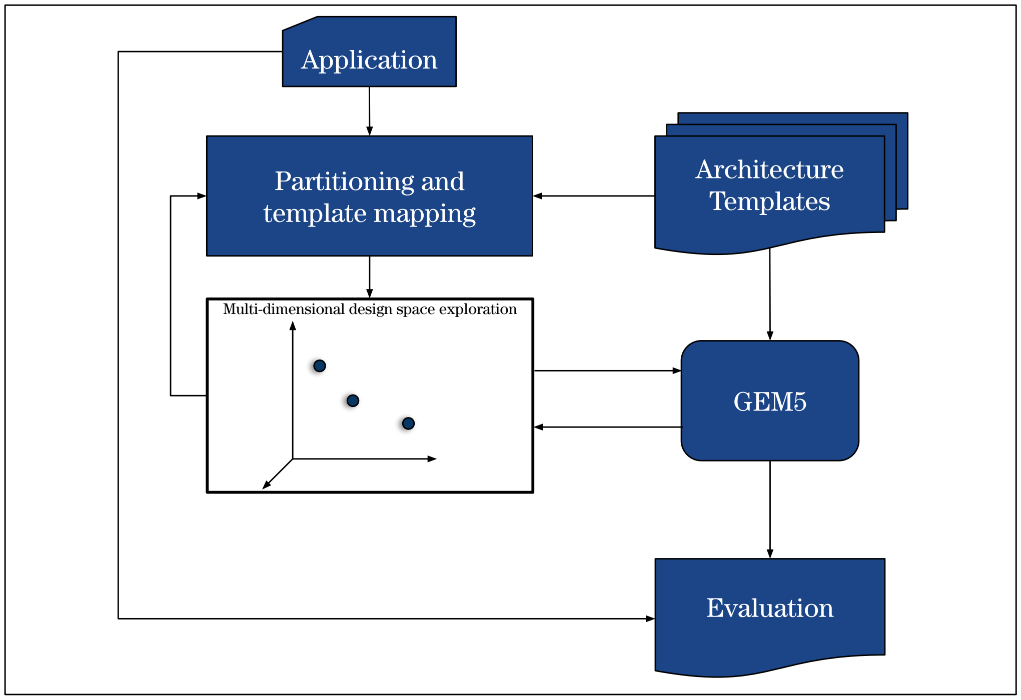

Application
Partitioning and template mapping
Architecture Templates
Multi-dimensional design space exploration
GEM5
Evaluation

Figure 8: Overview of the ANML Mapping Tools

.

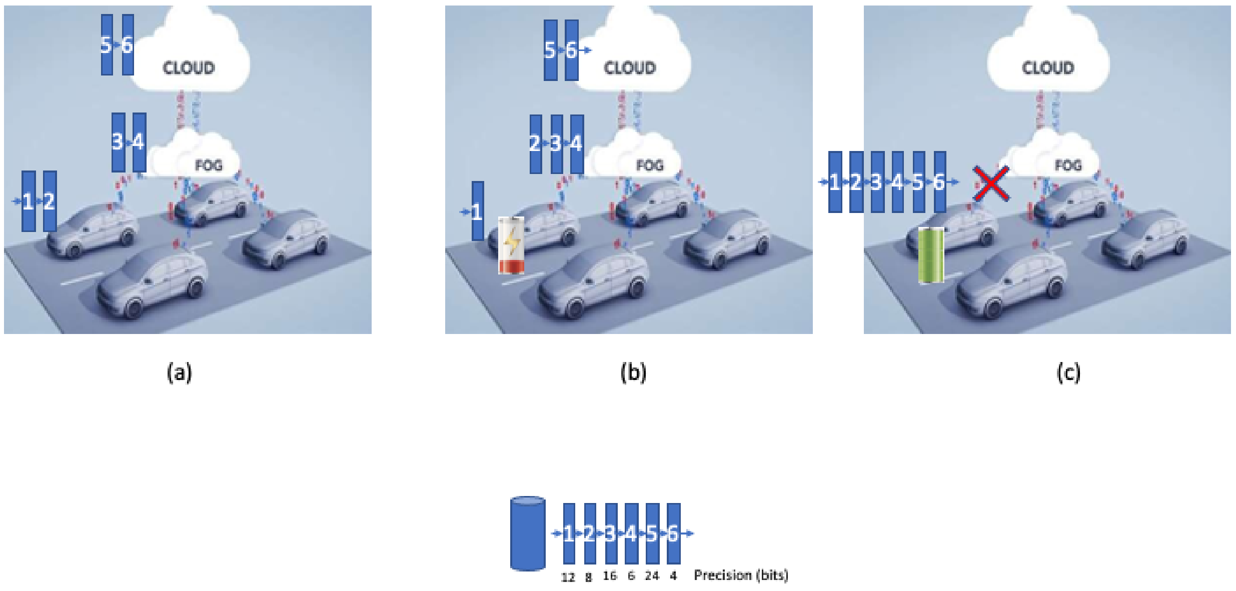


Figure 9: Dynamic Behavior is an Autonomous Vehicle ML - Part 1. The tasks are partitioned across the network. (a) balanced partitioning, (b) Vehicle's battery is nearly depleted so most of the computation is offloaded to the fog/cloud, (c) Communication link to the fog/cloud is unreliable so all the computation is performed in-vehicle

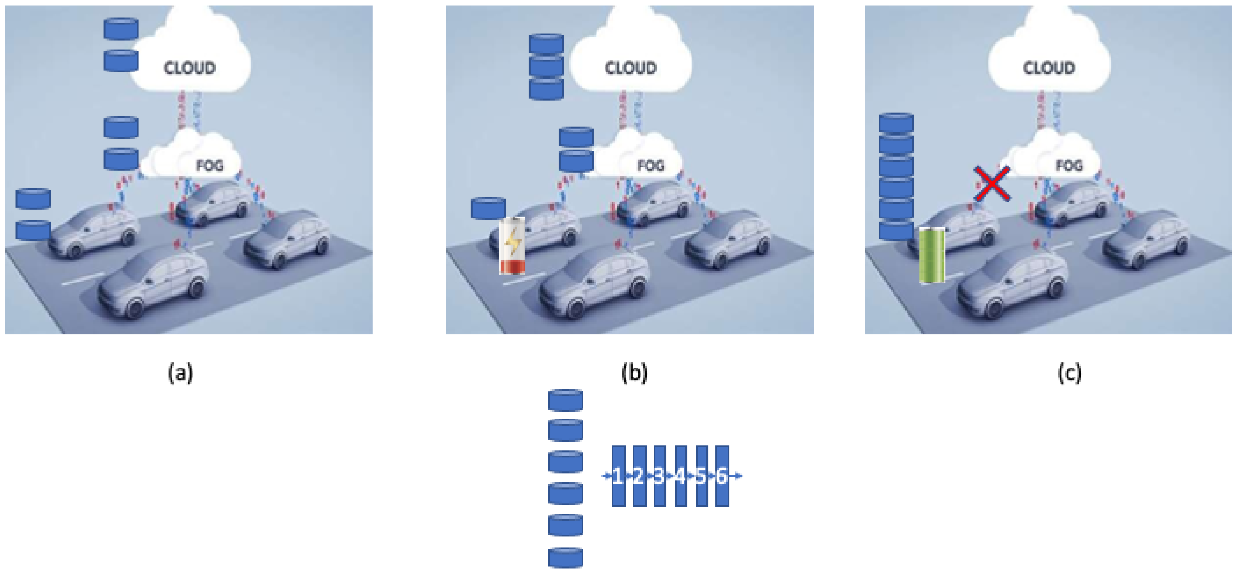


Figure 10: Dynamic Behavior is an Autonomous Vehicle ML - Part 2. Data is partitioned across the network. (a) balanced tiering across the network, (b)Vehicle's battery is nearly depleted so most of the data is processed in fog/cloud, (c)Communication link to the fog/cloud is unreliable so all the data is performed in-vehicle

References

[1] C. Zhang, P. Patras, and H. Haddadi, "Deep learning in mobile and wireless networking: A survey," *IEEE Communications Surveys Tutorials*, vol. 21, no. 3, pp. 2224–2287, 2019.

[2] T. Taleb, "On multi-access edge computing: A survey of the emerging 5g network edge cloud architecture and orchestration," *IEEE Commun. Surveys Tuts*, vol. 19, no. 3, pp. 1657–1681, 2017.

[3] Y. Mao, C. You, J. Zhang, K. Huang, and K. Letaief, "A survey on mobile edge computing: The communication perspective," *IEEE Commun. Surveys Tuts*, vol. 19, no. 4, pp. 2322–2358, 2017.

[4] R. Li, "Intelligent 5g: When cellular networks meet artificial intelligence," *IEEE Wireless Commun*, vol. 24, no. 5, pp. 175–183, Oct. 2017.

[5] M. Mohammadi, A. Al-Fuqaha, S. Sorour, and M. Guizani, "Deep learning for iot big data and streaming analytics: A survey," *IEEE Commun. Surveys Tuts*, vol. 20, no. 4, pp. 2923–2960, 2018.

[6] P. Mach and Z. Becvar, "Mobile edge computing: A survey on architecture and computation offloading," *IEEE Commun. Surveys Tuts*, vol. 19, no. 3, pp. 1628–1656, 2017.

[7] Y. Wang, "A data-driven architecture for personalized qoe management in 5g wireless networks," *IEEE Wireless Commun*, vol. 24, no. 1, pp. 102–110, Feb. 2017.

[8] T. Buda, "Can machine learning aid in delivering new use cases and scenarios in 5g?" in *Proc. IEEE/IFIP Netw. Oper. Manag. Symp. (NOMS)*, Istanbul, Turkey, 2016, pp. 1279–1284.

[9] R. A. Cooke and S. A. Fahmy, "A model for distributed in-network and near-edge computing with heterogeneous hardware," *Future Generation Computer Systems*, vol. 105, pp. 395–409, 2020.

[10] I. Lütkebohle, "BWorld Robot Control Software," https://images.nvidia.com/content/volta-architecture/pdf/volta-architecture-whitepaper.pdf, 2017, [Online; accessed April-2021].

[11] N. Jouppi, C. Young, N. Patil, and D. Patterson, "Motivation for and evaluation of the first tensor processing unit," *IEEE Micro*, vol. 38, no. 3, pp. 10–19, 2018.

[12] Y. Chen, T. Chen, Z. Xu, N. Sun, and O. Temam, "Diannao family: energy-efficient hardware accelerators for machine learning," *Communications of the ACM*, vol. 59, no. 11, pp. 105–112, 2016.

[13] A. Shafiee, A. Nag, N. Muralimanohar, R. Balasubramonian, J. P. Strachan, M. Hu, R. S. Williams, and V. Srikumar, "Isaac: A convolutional neural network accelerator with in-situ analog arithmetic in crossbars," *ACM SIGARCH Computer Architecture News*, vol. 44, no. 3, pp. 14–26, 2016.

[14] P. Chi, S. Li, C. Xu, T. Zhang, J. Zhao, Y. Liu, Y. Wang, and Y. Xie, "Prime: A novel processing-in-memory architecture for neural network computation in reram-based main memory," *ACM SIGARCH Computer Architecture News*, vol. 44, no. 3, pp. 27–39, 2016.

[15] S. Tang, S. Yin, S. Zheng, P. Ouyang, F. Tu, L. Yao, J. Wu, W. Cheng, L. Liu, and S. Wei, "Aepe: An area and power efficient rram crossbar-based accelerator for deep cnns," in *2017 IEEE 6th Non-Volatile Memory Systems and Applications Symposium (NVMSA)*. IEEE, 2017, pp. 1–6.

[16] T. Chou, W. Tang, J. Botimer, and Z. Zhang, "Cascade: Connecting rrams to extend analog dataflow in an end-to-end in-memory processing paradigm," in *Proceedings of the 52nd Annual IEEE/ACM International Symposium on Microarchitecture*, 2019, pp. 114–125.

[17] A. Ankit, I. E. Hajj, S. R. Chalamalasetti, G. Ndu, M. Foltin, R. S. Williams, P. Faraboschi, W.-m. W. Hwu, J. P. Strachan, K. Roy *et al.*, "PUMA: A programmable ultra-efficient memristor-based accelerator for machine learning inference," in *Proceedings of the Twenty-Fourth International Conference on Architectural Support for Programming Languages and Operating Systems*, 2019, pp. 715–731.

[18] L. Song, X. Qian, H. Li, and Y. Chen, "Pipelayer: A pipelined reram-based accelerator for deep learning," in *2017 IEEE International Symposium on High Performance Computer Architecture (HPCA)*. IEEE, 2017, pp. 541–552.

[19] X. Qiao, X. Cao, H. Yang, L. Song, and H. Li, "Atomlayer: a universal reram-based cnn accelerator with atomic layer computation," in *Proceedings of the 55th Annual Design Automation Conference*, 2018, pp. 1–6.

[20] A. Ankit, I. El Hajj, S. R. Chalamalasetti, S. Agarwal, M. Marinella, M. Foltin, J. P. Strachan, D. Milojicic, W.-M. Hwu, and K. Roy, "Panther: A programmable architecture for neural network training harnessing energy-efficient reram," *IEEE Transactions on Computers*, vol. 69, no. 8, pp. 1128–1142, 2020.

[21] Y. Gao, M. Kim, S. Abuadbba, Y. Kim, C. Thapa, K. Kim, S. A. Camtepe, H. Kim, and S. Nepal, "End-to-end evaluation of federated learning and split learning for internet of things," *arXiv preprint arXiv:2003.13376*, 2020.

[22] M. E. Fouda, A. M. Eltawil, and F. Kurdahi, "Modeling and analysis of passive switching crossbar arrays," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 65, no. 1, pp. 270–282, 2017.

[23] H. E. Yantir, A. M. Eltawil, and F. J. Kurdahi, "Approximate memristive in-memory computing," *ACM Transactions on Embedded Computing Systems (TECS)*, vol. 16, no. 5s, pp. 1–18, 2017.

[24] M. E. Fouda, A. M. Eltawil, and F. J. Kurdahi, "On one step row readout technique of selector-less resistive arrays," in *2017 IEEE 60th International Midwest Symposium on Circuits and Systems (MWSCAS)*. IEEE, 2017, pp. 72–75.

[25] M. A. Bahloul, M. E. Fouda, R. Naous, M. A. Zidan, A. M. Eltawil, F. Kurdahi, and K. N. Salama, "Design and analysis of 2t-2m ternary content addressable memories," in *2017 IEEE 60th International Midwest Symposium on Circuits and Systems (MWSCAS)*. IEEE, 2017, pp. 1430–1433.

[26] X. Chen, A. M. Eltawil, and F. J. Kurdahi, "Low latency approximate adder for highly correlated input streams," in *2017 IEEE International Conference on Computer Design (ICCD)*. IEEE, 2017, pp. 121–124.

[27] M. A. Neggaz, H. E. Yantır, S. Niar, A. Eltawil, and F. Kurdahi, "Rapid in-memory matrix multiplication using associative processor," in *2018 Design, Automation & Test in Europe Conference & Exhibition (DATE)*. IEEE, 2018, pp. 985–990.

[28] H. E. Yantır, A. M. Eltawil, and F. J. Kurdahi, "A two-dimensional associative processor," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 26, no. 9, pp. 1659–1670, 2018.

[29] ——, "A hybrid approximate computing approach for associative in-memory processors," *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, vol. 8, no. 4, pp. 758–769, 2018.

[30] R. A. Abdelaal, H. E. Yantır, A. M. Eltawil, and F. J. Kurdahi, "Power performance tradeoffs using adaptive bit width adjustments on resistive associative processors," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 66, no. 1, pp. 302–312, 2018.

[31] M. E. Fouda, E. Neftci, A. Eltawil, and F. Kurdahi, "Independent component analysis using RRAMs," *IEEE Transactions on Nanotechnology*, vol. 18, pp. 611–615, 2018.

[32] M. E. Fouda, J. Lee, A. M. Eltawil, and F. Kurdahi, "Overcoming crossbar nonidealities in binary neural networks through learning," in *Proceedings of the 14th IEEE/ACM International Symposium on Nanoscale Architectures*, 2018, pp. 31–33.

[33] M. Zorgui, M. E. Fouda, Z. Wang, A. Eltawil, and F. Kurdahi, "Polar coding for selector-less resistive memories," in *10th Annual Non-Volatile Memories Workshop*, 2019.

[34] M. E. Fouda, A. M. Eltawil, and F. Kurdahi, "Activated current sensing circuit for resistive neuromorphic networks," in *2019 17th IEEE International New Circuits and Systems Conference (NEWCAS)*. IEEE, 2019, pp. 1–4.

[35] M. Fouda, F. Kurdahi, A. Eltawil, and E. Neftci, "Spiking neural networks for inference and learning: A memristor-based design perspective," *arXiv preprint arXiv:1909.01771*, 2019.

[36] M. Payvand, M. E. Fouda, F. Kurdahi, A. Eltawil, and E. O. Neftci, "Error-triggered three-factor learning dynamics for crossbar arrays," in *2020 2nd IEEE International Conference on Artificial Intelligence Circuits and Systems (AICAS)*. IEEE, 2020, pp. 218–222.

[37] M. E. Fouda, S. Shaboyan, A. Elezabi, and A. Eltawil, "Application of ica on self-interference cancellation of in-band full duplex systems," *arXiv preprint arXiv:2001.00962*, 2020.

[38] H. E. Yantır, A. M. Eltawil, and K. N. Salama, "Efficient acceleration of stencil applications through in-memory computing," *Micromachines*, vol. 11, no. 6, p. 622, 2020.

[39] W. Guo, H. E. Yantır, M. E. Fouda, A. M. Eltawil, and K. N. Salama, "Towards efficient neuromorphic hardware: unsupervised adaptive neuron pruning," *Electronics*, vol. 9, no. 7, p. 1059, 2020.

[40] M. Rakka, M. Fouda, R. Kanj, A. Eltawil, and F. Kurdahi, “Design exploration of sensing techniques in 2t-2r resistive ternary cams,” *IEEE Transactions on Circuits and Systems II: Express Briefs*, 2020.

[41] M. Fouda, A. AbdelAty, A. Elwakil, A. Radwan, and A. Eltawil, “Programmable constant phase element realization with crossbar arrays,” *Journal of Advanced Research*, 2020.

[42] S. Arzykulov, A. Celik, G. Nauryzbayev, and A. M. Eltawil, “Uav-assisted cooperative & cognitive noma: Deployment, clustering, and resource allocation,” *arXiv preprint arXiv:2008.11356*, 2020.

[43] A. Ebrahim, A. Celik, E. Alsusa, and A. M. Eltawil, “Noma/oma mode selection and resource allocation for beyond 5g networks,” in *2020 IEEE 31st Annual International Symposium on Personal, Indoor and Mobile Radio Communications*. IEEE, 2020, pp. 1–6.

[44] S. Lee, G. Jung, M. E. Fouda, J. Lee, A. Eltawil, and F. Kurdahi, “Learning to predict ir drop with effective training for reram-based neural network hardware,” in *2020 57th ACM/IEEE Design Automation Conference (DAC)*. IEEE, 2020, pp. 1–6.

[45] W. Guo, M. E. Fouda, H. E. Yantir, A. M. Eltawil, and K. N. Salama, “Unsupervised adaptive weight pruning for energy-efficient neuromorphic systems,” *Frontiers in Neuroscience*, vol. 14, p. 1189, 2020.

[46] M. Payvand, M. E. Fouda, F. Kurdahi, A. M. Eltawil, and E. O. Neftci, “On-chip error-triggered learning of multi-layer memristive spiking neural networks,” *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, vol. 10, no. 4, pp. 522–535, 2020.

[47] M. E. Fouda, S. Lee, J. Lee, G. H. Kim, F. Kurdahi, and A. Eltawil, “Ir-qnn framework: An ir drop-aware offline training of quantized crossbar arrays,” *IEEE Access*, 2020.

[48] H. E. Yantır, A. M. Eltawil, and K. N. Salama, “Imca: An efficient in-memory convolution accelerator,” *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 29, no. 3, pp. 447–460, 2021.

[49] J. Bazzi, M. E. Fouda, R. Kanj, and A. M. Eltawil, “Threshold switch modeling for analog cam design,” in *2020 32nd International Conference on Microelectronics (ICM)*. IEEE, 2020, pp. 1–4.

[50] W. Guo, H. E. Yantir, M. E. Fouda, A. M. Eltawil, and K. N. Salama, “Toward the optimal design and fpga implementation of spiking neural networks,” *IEEE Transactions on Neural Networks and Learning Systems*, 2021.

[51] A. Abdallah, A. Celik, M. M. Mansour, and A. M. Eltawil, “Deep learning based frequency-selective channel estimation for hybrid mmwave mimo systems,” *arXiv preprint arXiv:2102.10847*, 2021.

[52] W. Guo, M. E. Fouda, A. M. Eltawil, and K. N. Salama, “Neural coding in spiking neural networks: A comparative study for robust neuromorphic systems,” *Frontiers in Neuroscience*, vol. 15, p. 212, 2021.

[53] B. Shihada, T. Elbatt, A. Eltawil, M. Mansour, E. Sabir, S. Rekhis, and S. Sharafeddine, “Networking research for the arab world: from regional initiatives to potential global impact,” *Communications of the ACM*, vol. 64, no. 4, pp. 114–119, 2021.

[54] S. Mondal, A. Eltawil, C.-A. Shen, and K. N. Salama, “Design and implementation of a sort-free k-best sphere decoder,” *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 18, no. 10, pp. 1497–1501, 2009.

[55] R. Naous, M. AlShedivat, E. Neftci, G. Cauwenberghs, and K. N. Salama, “Memristor-based neural networks: Synaptic versus neuronal stochasticity,” *AIP Advances*, vol. 6, no. 11, p. 111304, 2016.

[56] H. E. Yantir, *Efficient Acceleration of Computation Using Associative In-memory Processing.* eScholarship, University of California, 2018.

[57] S. Mondal, A. M. Eltawil, and K. N. Salama, “Architectural optimizations for low-power k-best mimo decoders,” *IEEE Transactions on Vehicular Technology*, vol. 58, no. 7, pp. 3145–3153, 2009.

[58] C.-A. Shen, A. M. Eltawil, K. N. Salama, and S. Mondal, “A best-first soft/hard decision tree searching mimo decoder for a 4x4 64-qam system,” *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 20, no. 8, pp. 1537–1541, 2011.

[59] M. E. Fouda, S. Lee, J. Lee, A. Eltawil, and F. Kurdahi, “Mask technique for fast and efficient training of binary resistive crossbar arrays,” *IEEE Transactions on Nanotechnology*, vol. 18, pp. 704–716, 2019. [Online]. Available: https://doi.org/10.1109%2Ftnano.2019.2927493

[60] A. E. Khorshid, I. N. Alquaydheb, F. Kurdahi, R. P. Jover, and A. Eltawil, "Biometric identity based on intra-body communication channel characteristics and machine learning," *Sensors*, vol. 20, no. 5, p. 1421, 2020.

[61] R. Cooke and S. Fahmy, "Exploring hardware accelerator offload for the internet of things," *IT - Information Technology*, vol. 62, no. 5-6, pp. 207–214, 2020, cited By 0. [Online]. Available: https://www.scopus.com/inward/record.uri?eid=2-s2.0-85097495932&doi=10.1515%2fitit-2020-0017&partnerID=40&md5=af362cf272c7c66af88406f248d1b792

[62] ——, "Characterizing latency overheads in the deployment of fpga accelerators," 2020, pp. 347–352, cited By 0. [Online]. Available: https://www.scopus.com/inward/record.uri?eid=2-s2.0-85095596279&doi=10.1109%2fFPL50879.2020.00064&partnerID=40&md5=027fb8276c26cd67b7f320c61b51460b

[63] L. Ioannou, A. Al-Dujaili, and S. Fahmy, "High throughput spatial convolution filters on fpgas," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 28, no. 6, pp. 1392–1402, 2020, cited By 0. [Online]. Available: https://www.scopus.com/inward/record.uri?eid=2-s2.0-85085945905&doi=10.1109%2fTVLSI.2020.2987202&partnerID=40&md5=231e2a4f0c4268136b299468a723393e

[64] R. Cooke and S. Fahmy, "Quantifying the latency benefits of near-edge and in-network fpga acceleration," 2020, pp. 7–12, cited By 2. [Online]. Available: https://www.scopus.com/inward/record.uri?eid=2-s2.0-85086592210&doi=10.1145%2f3378679.3394534&partnerID=40&md5=6065306aa9d7c9604ed1044e4801f17b

[65] ——, "A model for distributed in-network and near-edge computing with heterogeneous hardware," *Future Generation Computer Systems*, vol. 105, pp. 395–409, 2020, cited By 5. [Online]. Available: https://www.scopus.com/inward/record.uri?eid=2-s2.0-85076669982&doi=10.1016%2fj.future.2019.11.040&partnerID=40&md5=24c89bdfc93d36d429a347c4263fdaf2

[66] K. Herath, A. Prakash, S. Fahmy, and T. Srikanthan, "Power-efficient mapping of large applications on modern heterogeneous fpgas," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2020, cited By 0. [Online]. Available: https://www.scopus.com/inward/record.uri?eid=2-s2.0-85099091987&doi=10.1109%2fTCAD.2020.3047722&partnerID=40&md5=539f3e4bfcfafe1fbef79ed62ef0e43e

[67] A. Bucknall, S. Shreejith, and S. Fahmy, "Network enabled partial reconfiguration for distributed fpga edge acceleration," vol. 2019-December, 2019, pp. 259–262, cited By 1. [Online]. Available: https://www.scopus.com/inward/record.uri?eid=2-s2.0-85084924009&doi=10.1109%2fICFPT47387.2019.00042&partnerID=40&md5=1a612a1d49550e231dc4cb41226b3559

[68] L. Ioannou and S. Fahmy, "Lightweight programmable dsp block overlay for streaming neural network acceleration," vol. 2019-December, 2019, pp. 355–358, cited By 0. [Online]. Available: https://www.scopus.com/inward/record.uri?eid=2-s2.0-85084856607&doi=10.1109%2fICFPT47387.2019.00066&partnerID=40&md5=c1cca8a3c8896c90a6509e2dafee6874

[69] ——, "Neural network overlay using fpga dsp blocks," 2019, pp. 252–253, cited By 0. [Online]. Available: https://www.scopus.com/inward/record.uri?eid=2-s2.0-85075639531&doi=10.1109%2fFPL.2019.00048&partnerID=40&md5=2b79ae3580e84b09228ef3056377bc9d

[70] ——, "Network intrusion detection using neural networks on fpga socs," 2019, pp. 232–238, cited By 2. [Online]. Available: https://www.scopus.com/inward/record.uri?eid=2-s2.0-85075636250&doi=10.1109%2fFPL.2019.00043&partnerID=40&md5=b13cb68175dc153602d95834bb11f377

[71] S. Roberts, S. Wright, S. Fahmy, and S. Jarvis, "The power-optimised software envelope," *ACM Transactions on Architecture and Code Optimization*, vol. 16, no. 3, 2019, cited By 2. [Online]. Available: https://www.scopus.com/inward/record.uri?eid=2-s2.0-85067586528&doi=10.1145%2f3321551&partnerID=40&md5=423a4be43be179c7901b6023f380b2f0

[72] S. Fahmy, "Design abstraction for autonomous adaptive hardware systems on fpgas," 2018, pp. 142–147, cited By 0. [Online]. Available: https://www.scopus.com/inward/record.uri?eid=2-s2.0-85059938438&doi=10.1109%2fAHS.2018.8541489&partnerID=40&md5=41333015d823b11ffdb0655c74ed0c5a

[73] A. Kokaram, M. Manzke, M. Leeser, M. Blott, and S. Fahmy, "Welcome message," 2018, p. xv, cited By 0. [Online]. Available: https://www.scopus.com/inward/record.uri?eid=2-s2.0-85060311303&doi=10.1109%2fFPL.2018.00005&partnerID=40&md5=24a38e5c5efe3a80f40eef2fe34b0736

[74] S. Shreejith, R. Cooke, and S. Fahmy, "A smart network interface approach for distributed applications on xilinx zynq socs," 2018, pp. 186–190, cited By 7. [Online]. Available: https://www.scopus.com/inward/record.uri?eid=2-s2.0-85060273591&doi=10.1109%2fFPL.2018.00038&partnerID=40&md5=305bfe369928ed23a57be8c4c328df61

[75] K. Vipin and S. Fahmy, “Fpga dynamic and partial reconfiguration: A survey of architectures, methods, and applications,” *ACM Computing Surveys*, vol. 51, no. 4, 2018, cited By 52. [Online]. Available: https://www.scopus.com/inward/record.uri?eid=2-s2.0-85053256521&doi=10.1145%2f3193827&partnerID=40&md5=5f6ff714eb008d856e65c7db407979f0

[76] S. Shreejith, L. Mathew, V. Prasad, and S. Fahmy, “Efficient spectrum sensing for aeronautical ldacs using low-power correlators,” *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 26, no. 6, pp. 1183–1191, 2018, cited By 11. [Online]. Available: https://www.scopus.com/inward/record.uri?eid=2-s2.0-85043470426&doi=10.1109%2fTVLSI.2018.2806624&partnerID=40&md5=6f314035bcf07bec9cb2538fbce04a7b

[77] X. Li, A. Jain, D. Maskell, and S. Fahmy, “A time-multiplexed fpga overlay with linear interconnect,” vol. 2018-January, 2018, pp. 1075–1080, cited By 3. [Online]. Available: https://www.scopus.com/inward/record.uri?eid=2-s2.0-85048835560&doi=10.23919%2fDATE.2018.8342171&partnerID=40&md5=323863804c2fae65bb55925ffce7624b

[78] S. Shreejith and S. Fahmy, “Smart network interfaces for advanced automotive applications,” *IEEE Micro*, vol. 38, no. 2, pp. 72–80, 2018, cited By 3. [Online]. Available: https://www.scopus.com/inward/record.uri?eid=2-s2.0-85046025432&doi=10.1109%2fMM.2018.022071137&partnerID=40&md5=83beef19adc8adbf16dcbd74501574dd

[79] H. Singh, M.-H. Lee, G. Lu, F. J. Kurdahi, N. Bagherzadeh, and E. M. Chaves Filho, “MorphoSys: an integrated reconfigurable system for data-parallel and computation-intensive applications,” *IEEE transactions on computers*, vol. 49, no. 5, pp. 465–481, 2000.

[80] K. N. Salama, M. A. Zidan, F. J. Kurdahi, A. Eltawil, and H. E. Yantir, “Resistive content addressable memory based in-memory computation architecture,” Jul. 2 2019, uS Patent 10,340,007.

[81] H. E. Yantir, W. Guo, A. M. Eltawil, F. J. Kurdahi, and K. N. Salama, “An ultra-area-efficient 1024-point in-memory fft processor,” *Micromachines*, vol. 10, no. 8, p. 509, 2019.

[82] Y. Yang, S. Aïssa, A. M. Eltawil, and K. N. Salama, “An interference cancellation strategy for broadcast in hierarchical cell structure,” in *IEEE Global Communications Conference, GLOBECOM 2014, Austin, TX, USA, December 8-12, 2014*. IEEE, 2014, pp. 1792–1797. [Online]. Available: https://doi.org/10.1109/GLOCOM.2014.7037068

[83] C. Shen, A. M. Eltawil, K. N. Salama, and S. Mondal, “A best-first soft/hard decision tree searching MIMO decoder for a 4 $\times$ 4 64-qam system,” *IEEE Trans. Very Large Scale Integr. Syst.*, vol. 20, no. 8, pp. 1537–1541, 2012. [Online]. Available: https://doi.org/10.1109/TVLSI.2011.2159821

[84] C. Shen, A. M. Eltawil, and K. N. Salama, “Evaluation framework for k-best sphere decoders,” *Journal of Circuits, Systems, and Computers*, vol. 19, no. 5, pp. 975–995, 2010. [Online]. Available: https://doi.org/10.1142/S0218126610006554

[85] S. Mondal, A. M. Eltawil, C. Shen, and K. N. Salama, “Design and implementation of a sort-free k-best sphere decoder,” *IEEE Trans. Very Large Scale Integr. Syst.*, vol. 18, no. 10, pp. 1497–1501, 2010. [Online]. Available: https://doi.org/10.1109/TVLSI.2009.2025168

[86] C. Shen, A. M. Eltawil, S. Mondal, and K. N. Salama, “A best-first tree-searching approach for ML decoding in MIMO system,” in *International Symposium on Circuits and Systems (ISCAS 2010), May 30 - June 2, 2010, Paris, France*. IEEE, 2010, pp. 3533–3536. [Online]. Available: https://doi.org/10.1109/ISCAS.2010.5537825

[87] S. Mondal, A. M. Eltawil, and K. N. Salama, “Architectural optimizations for low-power -best MIMO decoders,” *IEEE Trans. Vehicular Technology*, vol. 58, no. 7, pp. 3145–3153, 2009. [Online]. Available: https://doi.org/10.1109/TVT.2009.2017548

[88] M. Al-Shedivat, R. Naous, G. Cauwenberghs, and K. N. Salama, “Memristors empower spiking neurons with stochasticity,” *IEEE J. Emerg. Sel. Topics Circuits Syst.*, vol. 5, no. 2, pp. 242–253, 2015. [Online]. Available: https://doi.org/10.1109/JETCAS.2015.2435512

[89] M. Al-Shedivat, R. Naous, E. Neftci, G. Cauwenberghs, and K. N. Salama, “Inherently stochastic spiking neurons for probabilistic neural computation,” in *7th International IEEE/EMBS Conference on Neural Engineering, NER 2015, Montpellier, France, April 22-24, 2015*. IEEE, 2015, pp. 356–359. [Online]. Available: https://doi.org/10.1109/NER.2015.7146633

[90] R. Naous, M. Al-Shedivat, E. Neftci, G. Cauwenberghs, and K. N. Salama, "Stochastic synaptic plasticity with memristor crossbar arrays," in *IEEE International Symposium on Circuits and Systems, ISCAS 2016, Montréal, QC, Canada, May 22-25, 2016*. IEEE, 2016, pp. 2078–2081. [Online]. Available: https://doi.org/10.1109/ISCAS.2016.7538988

[91] S. Dang, O. Amin, B. Shihada, and M.-S. Alouini, "What should 6g be?" *Nature Electronics*, vol. 3, no. 1, pp. 20–29, 2020.

[92] A. Celik, A. Chaaban, B. Shihada, and M.-S. Alouini, "Topology optimization for 6g networks: A network information-theoretic approach," *arXiv preprint arXiv:1912.11498*, 2019.

[93] A. Celik, N. Saeed, B. Shihada, T. Y. Al-Naffouri, and M.-S. Alouini, "End-to-end performance analysis of underwater optical wireless relaying and routing techniques under location uncertainty," *IEEE Transactions on Wireless Communications*, 2019.

[94] H. El Hammouti, M. Benjillali, B. Shihada, and M.-S. Alouini, "Learn-as-you-fly: A distributed algorithm for joint 3d placement and user association in multi-uavs networks," *IEEE Transactions on Wireless Communications*, vol. 18, no. 12, pp. 5831–5844, 2019.

[95] I. AlQerm and B. Shihada, "Energy efficient traffic offloading in multi-tier heterogeneous 5g networks using intuitive online reinforcement learning," *IEEE Transactions on Green Communications and Networking*, vol. 3, no. 3, pp. 691–702, 2019.

[96] S. Hassan, S. Attia, K. N. Salama, and H. Mostafa, "Eann: Energy adaptive neural networks," *Electronics*, vol. 9, no. 5, p. 746, 2020.

[97] A. J. A. El-Maksoud, Y. O. Elmasry, K. N. Salama, and H. Mostafa, "Asic oriented comparative analysis of biologically inspired neuron models," in *2018 IEEE 61st International Midwest Symposium on Circuits and Systems (MWSCAS)*. IEEE, 2018, pp. 504–507.

[98] H. Mostafa, V. Ramesh, and G. Cauwenberghs, "Deep supervised learning using local errors," *Frontiers in neuroscience*, vol. 12, p. 608, 2018.

[99] E. O. Neftci, H. Mostafa, and F. Zenke, "Surrogate gradient learning in spiking neural networks: Bringing the power of gradient-based optimization to spiking neural networks," *IEEE Signal Processing Magazine*, vol. 36, no. 6, pp. 51–63, 2019.

[100] A. Nøkland, "Direct feedback alignment provides learning in deep neural networks," in *Advances in neural information processing systems*, 2016, pp. 1037–1045.

[101] G. Detorakis, T. Bartley, and E. Neftci, "Contrastive hebbian learning with random feedback weights," *Neural Networks*, vol. 114, pp. 1–14, 2019.

[102] D. Ielmini and H.-S. P. Wong, "In-memory computing with resistive switching devices," *Nature Electronics*, vol. 1, no. 6, pp. 333–343, 2018.

[103] F. Akopyan, J. Sawada, A. Cassidy, R. Alvarez-Icaza, J. Arthur, P. Merolla, N. Imam, Y. Nakamura, P. Datta, G.-J. Nam *et al.*, "TrueNorth: Design and tool flow of a 65 mW 1 million neuron programmable neurosynaptic chip," *IEEE transactions on computer-aided design of integrated circuits and systems*, vol. 34, no. 10, pp. 1537–1557, 2015.

[104] M. Davies, N. Srinivasa, T.-H. Lin, G. Chinya, Y. Cao, S. H. Choday, G. Dimou, P. Joshi, N. Imam, S. Jain *et al.*, "Loihi: A neuromorphic manycore processor with on-chip learning," *IEEE Micro*, vol. 38, no. 1, pp. 82–99, 2018.

[105] X. Si, Y.-N. Tu, W.-H. Huanq, J.-W. Su, P.-J. Lu, J.-H. Wang, T.-W. Liu, S.-Y. Wu, R. Liu, Y.-C. Chou *et al.*, "15.5 a 28nm 64kb 6t sram computing-in-memory macro with 8b mac operation for ai edge chips," in *2020 IEEE International Solid-State Circuits Conference-(ISSCC)*. IEEE, 2020, pp. 246–248.

[106] M. E. Sinangil, B. Erbagci, R. Naous, K. Akarvardar, D. Sun, W.-S. Khwa, H.-J. Liao, Y. Wang, and J. Chang, "A 7-nm compute-in-memory sram macro supporting multi-bit input, weight and output and achieving 351 tops/w and 372.4 gops," *IEEE Journal of Solid-State Circuits*, vol. 56, no. 1, pp. 188–198, 2020.

[107] H. Liu, M. Liu, Z. Zhu, and Y. Yang, "A high linear voltage-to-time converter (vtc) with 1.2 v input range for time-domain analog-to-digital converters," *Microelectronics Journal*, vol. 88, pp. 18–24, 2019.

[108] A. El-Bayoumi, H. Mostafa, and A. M. Soliman, “A novel mim-capacitor-based 1-gs/s 14-bit variation-tolerant fully-differential voltage-to-time converter (vtc) circuit,” *Journal of Circuits, Systems and Computers*, vol. 26, no. 05, p. 1750073, 2017.

[109] A. Elgreatly, A. Dessouki, H. Mostafa, R. Abdalla, and E.-s. El-Rabaie, “A novel highly linear voltage-to-time converter (vtc) circuit for time-based analog-to-digital converters (adc) using body biasing,” *Electronics*, vol. 9, no. 12, p. 2033, 2020.

[110] J. Kim, C. Lee, J. Kim, Y. Kim, C. S. Hwang, and K. Choi, “Vcam: Variation compensation through activation matching for analog binarized neural networks,” in *2019 IEEE/ACM International Symposium on Low Power Electronics and Design (ISLPED)*. IEEE, 2019, pp. 1–6.

[111] M. E. Fouda, E. Neftci, A. Eltawil, and F. Kurdahi, “Effect of asymmetric nonlinearity dynamics in RRAMs on spiking neural network performance,” in *2019 53rd Asilomar Conference on Signals, Systems, and Computers*. IEEE, 2019, pp. 495–499.

[112] M. Payvand, M. Fouda, F. Kurdahi, A. Eltawil, and E. O. Neftci, “Error-triggered three-factor learning dynamics for crossbar arrays,” *arXiv preprint arXiv:1910.06152*, 2019.

[113] E. O. Neftci, C. Augustine, S. Paul, and G. Detorakis, “Event-driven random back-propagation: Enabling neuromorphic deep learning machines,” *Frontiers in neuroscience*, vol. 11, p. 324, 2017.

[114] J. Park, T. Yu, S. Joshi, C. Maier, and G. Cauwenberghs, “Hierarchical address event routing for reconfigurable large-scale neuromorphic systems,” *IEEE transactions on neural networks and learning systems*, vol. 28, no. 10, pp. 2408–2422, 2016.

[115] V. Kornijcuk, J. Park, G. Kim, D. Kim, I. Kim, J. Kim, J. Y. Kwak, and D. S. Jeong, “Reconfigurable spike routing architectures for on-chip local learning in neuromorphic systems,” *Advanced Materials Technologies*, vol. 4, no. 1, p. 1800345, 2019.

[116] M. Palesi and M. Daneshtalab, *Routing algorithms in networks-on-chip*. Springer, 2014.

[117] N. Binkert, B. Beckmann, G. Black, S. K. Reinhardt, A. Saidi, A. Basu, J. Hestness, D. R. Hower, T. Krishna, S. Sardashti, R. Sen, K. Sewell, M. Shoaib, N. Vaish, M. D. Hill, and D. A. Wood, “The gem5 simulator,” *SIGARCH Comput. Archit. News*, vol. 39, no. 2, p. 1–7, Aug. 2011. [Online]. Available: https://doi.org/10.1145/2024716.2024718

[118] S. Li, J. H. Ahn, R. D. Strong, J. B. Brockman, D. M. Tullsen, and N. P. Jouppi, “Mcpat: An integrated power, area, and timing modeling framework for multicore and manycore architectures,” in *Proceedings of the 42nd Annual IEEE/ACM International Symposium on Microarchitecture*, ser. MICRO 42. New York, NY, USA: Association for Computing Machinery, 2009, p. 469–480. [Online]. Available: https://doi.org/10.1145/1669112.1669172

[119] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, M. Kudlur, J. Levenberg, R. Monga, S. Moore, D. G. Murray, B. Steiner, P. Tucker, V. Vasudevan, P. Warden, M. Wicke, Y. Yu, and X. Zheng, “Tensorflow: A system for large-scale machine learning,” in *Proceedings of the 12th USENIX Conference on Operating Systems Design and Implementation*, ser. OSDI’16. USA: USENIX Association, 2016, p. 265–283.

[120] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Köpf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, “Pytorch: An imperative style, high-performance deep learning library,” 2019.

[121] C. Hao, X. Zhang, Y. Li, S. Huang, J. Xiong, K. Rupnow, W. Hwu, and D. Chen, “Fpga/dnn co-design: An efficient design methodology for 1ot intelligence on the edge,” in *2019 56th ACM/IEEE Design Automation Conference (DAC)*, 2019, pp. 1–6.

[122] J. Zeppenfeld, A. Bouajila, W. Stechele, and A. Herkersdorf, “Learning classifier tables for autonomic systems on chip,” in *INFORMATIK 2008. Beherrschbare Systeme - dank Informatik. Band 2*, H.-G. Hegering, A. Lehmann, H. J. Ohlbach, and C. Scheideler, Eds. Bonn: Gesellschaft für Informatik e. V., 2008, pp. 771–778.

[123] M. Möstl, J. Schlatow, R. Ernst, N. Dutt, A. Nassar, A. Rahmani, F. J. Kurdahi, T. Wild, A. Sadighi, and A. Herkersdorf, “Platform-centric self-awareness as a key enabler for controlling changes in cps,” *Proceedings of the IEEE*, vol. 106, no. 9, pp. 1543–1567, 2018.

[124] M. Behrisch, L. Bieker, J. Erdmann, and D. Krajzewicz, "Sumo–simulation of urban mobility: an overview," in *Proceedings of SIMUL 2011, The Third International Conference on Advances in System Simulation.* ThinkMind, 2011.

[125] M. Seo and F. Kurdahi, "Efficient tracing methodology using automata processor," *ACM Trans. Embed. Comput. Syst.*, vol. 18, no. 5s, Oct. 2019. [Online]. Available: https://doi.org/10.1145/3358200

[126] M. Seo and R. Lysecky, "Non-intrusive in-situ requirements monitoring of embedded system," *ACM Transactions on Design Automation of Electronic Systems (TODAES)*, vol. 23, no. 5, pp. 1–27, 2018.

[127] M. A. Bahloul, M. E. Fouda, R. Naous, M. A. Zidan, A. M. Eltawil, F. J. Kurdahi, and K. N. Salama, "Design and analysis of 2t-2m ternary content addressable memories," in *IEEE 60th International Midwest Symposium on Circuits and Systems, MWSCAS 2017, Boston, MA, USA, August 6-9, 2017.* IEEE, 2017, pp. 1430–1433. [Online]. Available: https://doi.org/10.1109/MWSCAS.2017.8053201

[128] W.-H. Chen, K.-X. Li, W.-Y. Lin, K.-H. Hsu, P.-Y. Li, C.-H. Yang, C.-X. Xue, E.-Y. Yang, Y.-K. Chen, Y.-S. Chang *et al.*, "A 65nm 1mb nonvolatile computing-in-memory reram macro with sub-16ns multiply-and-accumulate for binary dnn ai edge processors," in *2018 IEEE International Solid-State Circuits Conference-(ISSCC).* IEEE, 2018, pp. 494–496.

[129] C.-X. Xue, W.-H. Chen, J.-S. Liu, J.-F. Li, W.-Y. Lin, W.-E. Lin, J.-H. Wang, W.-C. Wei, T.-Y. Huang, T.-W. Chang *et al.*, "Embedded 1-mb reram-based computing-in-memory macro with multibit input and weight for cnn-based ai edge processors," *IEEE Journal of Solid-State Circuits*, vol. 55, no. 1, pp. 203–215, 2019.

[130] C.-X. Xue, T.-Y. Huang, J.-S. Liu, T.-W. Chang, H.-Y. Kao, J.-H. Wang, T.-W. Liu, S.-Y. Wei, S.-P. Huang, W.-C. Wei *et al.*, "15.4 a 22nm 2mb reram compute-in-memory macro with 121-28tops/w for multibit mac computing for tiny ai edge devices," in *2020 IEEE International Solid-State Circuits Conference-(ISSCC).* IEEE, 2020, pp. 244–246.

[131] Q. Liu, B. Gao, P. Yao, D. Wu, J. Chen, Y. Pang, W. Zhang, Y. Liao, C.-X. Xue, W.-H. Chen *et al.*, "33.2 a fully integrated analog reram based 78.4 tops/w compute-in-memory chip with fully parallel mac computing," in *2020 IEEE International Solid-State Circuits Conference-(ISSCC).* IEEE, 2020, pp. 500–502.

[132] Y. Chen, L. Lu, B. Kim, and T. T.-H. Kim, "A reconfigurable 4t2r reram computing in-memory macro for efficient edge applications," *IEEE Open Journal of Circuits and Systems*, vol. 2, pp. 210–222, 2021.

[133] J. Sim, J.-S. Park, M. Kim, D. Bae, Y. Choi, and L.-S. Kim, "14.6 a 1.42 tops/w deep convolutional neural network recognition processor for intelligent ioe systems," in *2016 IEEE International Solid-State Circuits Conference (ISSCC).* IEEE, 2016, pp. 264–265.

[134] W.-S. Khwa, J.-J. Chen, J.-F. Li, X. Si, E.-Y. Yang, X. Sun, R. Liu, P.-Y. Chen, Q. Li, S. Yu *et al.*, "A 65nm 4kb algorithm-dependent computing-in-memory sram unit-macro with 2.3 ns and 55.8 tops/w fully parallel product-sum operation for binary dnn edge processors," in *2018 IEEE International Solid-State Circuits Conference-(ISSCC).* IEEE, 2018, pp. 496–498.

[135] A. Biswas and A. P. Chandrakasan, "Conv-ram: An energy-efficient sram with embedded convolution computation for low-power cnn-based machine learning applications," in *2018 IEEE International Solid-State Circuits Conference-(ISSCC).* IEEE, 2018, pp. 488–490.

[136] X. Si, J.-J. Chen, Y.-N. Tu, W.-H. Huang, J.-H. Wang, Y.-C. Chiu, W.-C. Wei, S.-Y. Wu, X. Sun, R. Liu *et al.*, "24.5 a twin-8t sram computation-in-memory macro for multiple-bit cnn-based machine learning," in *2019 IEEE International Solid-State Circuits Conference-(ISSCC).* IEEE, 2019, pp. 396–398.

[137] H. Kim, Q. Chen, T. Yoo, T. T.-H. Kim, and B. Kim, "A 1-16b precision reconfigurable digital in-memory computing macro featuring column-mac architecture and bit-serial computation," in *ESSCIRC 2019-IEEE 45th European Solid State Circuits Conference (ESSCIRC).* IEEE, 2019, pp. 345–348.

[138] Y.-D. Chih, P.-H. Lee, H. Fujiwara, Y.-C. Shih, C.-F. Lee, R. Naous, Y.-L. Chen, C.-P. Lo, C.-H. Lu, H. Mori *et al.*, "An 89tops/w and 16.3 tops/mm 2 all-digital sram-based full-precision compute-in memory macro in 22nm for machine-learning edge applications," in *2021 IEEE International Solid-State Circuits Conference (ISSCC)*, vol. 64. IEEE, 2021, pp. 252–254.

Biographical Information Key Personnel

Name	Title	Institution/Company
Ahmed M. Eltawil	Professor	King Abdullah University of Science and Technology

Website URL	Country
https://cemse.kaust.edu.sa/ccsl	Saudi Arabia

Education and Employment Positions

EDUCATION

Ph.D.	2003	***University of California, Los Angeles.*** A universal RAKE receiver for DS-CDMA broadband wireless communications	Integrated Circuits and systems
M.Sc.	1999	***Cairo University, Cairo, Egypt.*** VLSI circuits implementing CDMA based wireless systems	Electronics and Communication Engineering
BS.	1997	***Cairo University, Cairo, Egypt.***	Electronics and Communication Engineering

ACADEMIC APPOITMENTS

KING ABDULLAH UNIVERSITY OF SCIENCE AND TECHNOLOGY (KAUST)

Computer, Electrical and Mathematical Sciences and Engineering (CEMSE) Division

Founder and Director of the Communication and Computing Systems Laboratory (CCSL)

08/2019 – Present: Full Professor

UNIVERSITY OF CALIFORNIA, IRVINE

Department of Electrical Engineering and Computer Science

Founder and Director of the Wireless Systems and Circuits Laboratory (WSCL)

07/2015 – 08/2019: Full Professor

07/2010 – 06/2015: Associate Professor

01/2005 – 06/2010: Assistant Professor, Henry Samueli Faculty Fellow

Center for Embedded and Cyber-Physical Systems (CECS)

01/2017 – 08/2019: Director of the Master in Embedded and Cyber Physical Systems (MECPS).

UNIVERSITY OF CALIFORNIA, LOS ANGELES

Department of Electrical Engineering

09/2003 – 01/2005: Project Scientist

INDUSTRIAL EXPERIENCE

EXPERT CONSULTANT, CA

09/2003 to Present: Consultant to several companies including TransformX, Intellisense Broadcom, Interdigital, Emulex, among others. Provided strategic vision to clients regarding leveraging state of the art technologies in computing and communication systems towards future products.

LEXTRUM INC., IRVINE, CA

10/2015 to 11/2017, **FOUNDER**

Lextrum developed Full Duplex solutions for 5G networks. Full Duplex is a technology that allows communication devices to listen and broadcast simultaneously, on the same frequency, thus effectively doubling spectrum utilization. FD alleviates the need for complicated network planning, by allowing the instantaneous exchange of network information between users, thus reducing collisions and improving throughput. Lextrum was acquired by TransformX (now ComSoverign), in 2019

NEWPORT MEDIA, LAKE FOREST, CA

01/2005 to 01/2010, **PRINCIPLE CONSULTANT TO FOUNDING TEAM**

Worked with the founding team to interview and hire core digital chip team. Together with both the system and chip leads, responsible for setting company strategy and vision for product lines. Newport Media was acquired by Atmel in 2014 as a supplier of connectivity solutions including 802.11n Wi-Fi and Bluetooth SoCs.

SILVUS TECHNOLOGIES, LOS ANGELES, CA

08/2003 to 12/2005, **FOUNDING PARTNER**

Silvus Technologies is a privately held firm that has become an established leader in MIMO radios. Silvus was the sole performer on the multi-year Defence Advanced Research Projects Agency's Mobile Networked MIMO (MNM) program. Together with the founders, I helped set the strategic vision of the company when it was founded and developed the first prototype MIMO radios.

INNOVICS WIRELESS, LOS ANGELES, CA

01/2001 to 08/2003, **MEMBER OF FOUNDING TEAM AND DIRECTOR OF VLSI ENGINEERING**

Founding member of a venture capital backed semiconductors company (Innovics Wireless), where the team delivered the first reported diversity enabled third generation W-CDMA mobile transceiver system on a chip.

Honors and Awards

- Elected **Senior Member of the National Academy of Inventors, 2021**.
- One of five nominees for the **2020 UCI Innovator of the year award** out of 1685 faculty considered at the University of California, Irvine.
- **Best Paper Award,** 3rd place, IEEE International Conference on Microelectronics, ICM 2020.
- 2016 **Provost's Leadership Academy,** recognizing faculty who show promise as future school leaders.
- Recommended as a **Fulbright Scholar** by the Council for International Exchange of Scholars, 2015.
- **Recipient of Honor award** from Salman Bin Abdulaziz University (Renamed to Prince Sattam Bin Abdulaziz University), Riyadh, Saudi Arabia, 2013.
- **Boeing Distinguished External Researcher and Scholar**, 2010.
- Recipient of the **National Science Foundation CAREER** award, 2010.
- Nominated for **Engineering Professor of the Year** Award by the UCI student council (2006/07, 2008/2009).
- **Henry Samueli Faculty Fellow** from January 2005 through December 2007.
- Recipient of the **Henry Samueli Excellence in Teaching Award**, School of Engineering, University of California, Los Angeles, 2001.

Research Interests

General area of signal processing architectures with an emphasis on mobile computing and communication systems and their applications spanning wireless networks, personal networks and Cyber-physical systems (CPS).

Five most recent publications	Total nº of publications: 210

Google Scholar: https://scholar.google.com/citations?user=XzW-KWoAAAAJ&hl=en

1. B. Shihada et al., "Networking research for the Arab world: from regional initiatives to potential global impact," Communications of the ACM, vol. 64, no. 4, pp. 114–119, 2021.
2. W. Guo, H. E. Yantir, M. E. Fouda, A. M. Eltawil, and K. N. Salama, "Toward the Optimal Design and FPGA Implementation of Spiking Neural Networks," IEEE Transactions on Neural Networks and Learning Systems, 2021.
3. S. Shaboyan, A. S. Behbahani, and A. M. Eltawil, "Design and Implementation of an End-to-End Amplify and Forward Full-Duplex Relay Network," IEEE Access, vol. 8, pp. 163594–163607, 2020.
4. M. E. Fouda, S. Lee, J. Lee, G. H. Kim, F. Kurdahi, and A. Eltawil, "IR-QNN Framework: An IR Drop-Aware Offline Training Of Quantized Crossbar Arrays," IEEE Access, 2020.
5. J. Bazzi, M. E. Fouda, R. Kanj, and A. M. Eltawil, "Threshold Switch Modeling for Analog CAM Design," in 2020 32nd International Conference on Microelectronics (ICM), 2020, pp. 1–4.**(Best Paper Award)**

Five most relevant publications to this proposal (if different from above)

1. Y. Na, S. El-Tawil, A. Ibrahim, and A. Eltawil, "Stick-Slip Classification Based on Machine Learning Techniques for Building Damage Assessment," Journal of Earthquake Engineering, pp. 1–18, 2021.
2. A. Abdallah, A. Celik, M. M. Mansour, and A. M. Eltawil, "Deep Learning Based Frequency-Selective Channel Estimation for Hybrid mmWave MIMO Systems," arXiv preprint arXiv:2102.10847, 2021.
3. M. Payvand, M. E. Fouda, F. Kurdahi, A. Eltawil, and E. O. Neftci, "Error-triggered three-factor learning dynamics for crossbar arrays," in 2020 2nd IEEE International Conference on Artificial Intelligence Circuits and Systems (AICAS), 2020, pp. 218–222.
4. W. Guo, H. E. Yantır, M. E. Fouda, A. M. Eltawil, and K. N. Salama, "Towards efficient neuromorphic hardware: unsupervised adaptive neuron pruning," Electronics, vol. 9, no. 7, p. 1059, 2020.
5. A. Ibrahim, A. Eltawil, Y. Na, and S. El-Tawil, "A machine learning approach for structural health monitoring using noisy data sets," IEEE Transactions on Automation Science and Engineering, vol. 17, no. 2, pp. 900–908, 2019.

Biographical Information Key Personnel

Page limit: 2 maximum   Duplicate the page for additional template tables as needed.

Name	Title	Institution/Company
Khaled Nabil Salama	Professor	KAUST
Website URL		**Country**
Sensors.kaust.edu.sa		Saudi Arabia

Education and Employment Positions

Education:
- Ph.D. in Electrical Engineering, Stanford University, Palo Alto, CA, USA, 2000-2005
- MSc. in Electrical Engineering, Stanford University, Palo Alto, CA, USA, 1998-2000
- BSc. in Electronic and Communications Engineering ,Cairo University, Cairo, Egypt, 1992-1997

Employment
- 2021-Present, Associate Dean, CMSE Division, KAUST
- 2018- Present Professor, CEMSE Division, KAUST
- 2013- 2017 Associate Professor, CEMSE Division, KAUST
- 2009- 2013 Assistant Professor, CEMSE Division, KAUST
- 2007-2008 Cofounder and CTO Ultrawave Labs, California USA
- 2005- 2009 Assistant Professor, Rensselaer Polytechnic Institute (RPI), NY, USA
- 1997-1998 Lecturer, Cairo University, Egypt
- 1997-1998 Engineer, Mentor Graphics, Cairo, Egypt

Honors and Awards

Stanford-Berkeley Innovators challenge award (biomedical track)
IEEE senior member
Best poster award 2019 Joint MMM-INTERMAG Conference held in Washington, D.C.

Research Interests

Brain inspired computing, memristors, integrated circuits and systems, neuromorphic computing.

Five most recent publications	Total nº of publications: 305

Publications: https://scholar.google.com/citations?user=wRUKAMMAAAAJ&hl=en&oi=ao

1. Naous, Rawan, Anne Siemon, Michael Schulten, Hamzah Alahmadi, Andreas Kindsmüller, Michael Lübben, Arne Heittmann, Rainer Waser, Khaled Nabil Salama, and Stephan Menzel. "Theory and experimental verification of configurable computing with stochastic memristors." *Scientific reports* 11, no. 1 (2021): 1-11.
2. Guo, Wenzhe, Mohammed E. Fouda, Ahmed M. Eltawil, and Khaled Nabil Salama. "Neural Coding in Spiking Neural Networks: A Comparative Study for Robust Neuromorphic Systems." *Frontiers in Neuroscience* 15 (2021): 212.
3. Guo, Wenzhe, Hasan Erdem Yantir, Mohammed E. Fouda, Ahmed M. Eltawil, and Khaled Nabil Salama. "Toward the Optimal Design and FPGA Implementation of Spiking Neural Networks." *IEEE Transactions on Neural Networks and Learning Systems* (2021).
4. Yantir, Hasan Erdem, Ahmed M. Eltawil, and Khaled N. Salama. "IMCA: An Efficient In-Memory Convolution Accelerator." *IEEE Transactions on Very Large Scale Integration (VLSI) Systems* 01 (1900): 1-14, 2021
5. Naous, Rawan, and Khaled Nabil Salama. "Stochastic memristor logic devices." U.S. Patent Application 16/922,255, filed October 22, 2020.
6.

Five most relevant publications to this proposal (if different from above)

1. Guo, Wenzhe, Mohammed E. Fouda, Hasan Erdem Yantir, Ahmed M. Eltawil, and Khaled Nabil Salama. "Unsupervised Adaptive Weight Pruning for Energy-Efficient Neuromorphic Systems." *Frontiers in Neuroscience* 14 (2020): 1189.
2. Krestinskaya, O., K. Salama, and A. P. James. "Towards Hardware Optimal Neural Network Selection with Multi-Objective Genetic Search." In *2020 IEEE International Symposium on Circuits and Systems (ISCAS)*, pp. 1-5. IEEE, 2020.
3. Guo, Wenzhe, Hasan Erdem Yantır, Mohammed E. Fouda, Ahmed M. Eltawil, and Khaled Nabil Salama. "Towards efficient neuromorphic hardware: unsupervised adaptive neuron pruning." *Electronics* 9, no. 7 (2020): 1059.
4. Yantir, Hasan Erdem, Wenzhe Guo, Ahmed M. Eltawil, Fadi J. Kurdahi, and Khaled Nabil Salama. "An ultra-area-efficient 1024-point in-memory fft processor." *Micromachines* 10, no. 8 (2019): 509.
5. Krestinskaya, Olga, Khaled N. Salama, and Alex P. James. "Automating analogue AI chip design with genetic search." *Advanced Intelligent Systems* 2, no. 8 (2020): 2000075.

Name	Title	Institution/Company
Suhaib Fahmy	Associate Professor	KAUST

Website URL	Country
https://accl.kaust.edu.sa/	Saudi Arabia

Education and Employment Positions

2020–now: Associate Professor, Computer Science, CEMSE, KAUST

2019–2020 : Reader in Computer Engineering, School of Engineering, University of Warwick, UK

2015–2019: Associate Professor, School of Engineering, University of Warwick, UK

2009–2015: Assistant Professor, School of Computer Engineering, Nanyang Technological University, Singapore

2007–2009: Postdoctoral Research Fellow, CTVR, Trinity College Dublin, Ireland

2007–2009: Visiting Research Engineer, Xilinx Research Labs, Ireland

2003–2007: PhD, Electrical and Electronic Engineering, Imperial College London, UK

1999–2003: MEng Information Systems Engineering (integrated Bachelors and Masters), 1st Class, Imperial College London, UK

Honors and Awards

2020: Royal Academy of Engineering/The Leverhulme Trust Research Fellowship

2019: Best Paper Award, ACM Transactions on Design Automation of Electronic Systems

2017–present: Turing Fellow, The Alan Turing Institute

2017: IBM Faculty Award

2016: Community Award, International Conference on Field Programmable Logic and Applications

2013: IBM Faculty Award

2012: Best Paper Award, International Conference on Field Programmable Technology

Advised 2 PostDocs, 9 PhD students, 3 research Masters students. After graduation, students have found employment as faculty at Trinity College Dublin, Nazarbayev University, BITS Pilani, India, and industry positions at Xilinx, Arm, Google, Audi and multiple startups.

Research Interests

- Hardware acceleration using FPGAs
- Accelerator interfacing and virtualisation
- Efficient machine learning architectures

Five most recent publications	**Total nº of publications: 103**
1. K. Herath, A. Prakash, S. A. Fahmy, and T. Srikanthan, Power-Efficient Mapping of Large Applications on Modern Heterogeneous FPGAs, IEEE TCAD, 2021. 2. K. Kamalakkannan, G. R. Mudalige, I. Z. Reguly, S. A Fahmy, High-Level FPGA Accelerator Design for Structured-Mesh-Based Explicit Numerical Solvers, Proceedings of IPDPS 2021. 3. A. R. Bucknall and S. A. Fahmy, Runtime Abstraction for Autonomous Adaptive Systems on Reconfigurable Hardware, Proceedings of DATE 2021. 4. I. Wardana, J. W. Gardner, S.A. Fahmy, Optimising Deep Learning at the Edge for Accurate Hourly Air Quality Prediction, Sensors 21 (4), 2021. 5. R. A. Cooke and S. A. Fahmy, Characterizing Latency Overheads in the Deployment of FPGA Accelerators, Proceedings of FPL 2020.	
Five most relevant publications to this proposal (if different from above)	
1. R. A. Cooke and S. A. Fahmy, Quantifying the Latency Benefits of Near-Edge and In-Network FPGA Acceleration, Proceedings of EdgeSys 2020. 2. S. Shreejith and S. A. Fahmy, Smart Network Interfaces for Advanced Automotive Applications, IEEE Micro, vol. 38, no. 2, 2018. 3. A. R. Bucknall, S. Shreejith, S. A. Fahmy, Network Enabled Partial Reconfiguration for Distributed FPGA Edge Acceleration, Proceedings of Field Programmable Technology, 2019. 4. S. A. Fahmy, K. Vipin, S. Shreejith, Virtualized FPGA Accelerators for Efficient Cloud Computing, Proceedings of CloudCom 2015. 5. R. A. Cooke and S. A. Fahmy, A Model for Distributed In-Network and Near-Edge Computing With Heterogeneous Hardware, Future Generation Computer Systems, vol. 105, April 2020.	

Biographical Information Key Personnel

Page limit: 2 maximum Duplicate the page for additional template tables as needed.

Name	**Title**	**Institution/Company**
Fadi Kurdahi	Professor	University of California, Irvine

Website URL	**Country**
http://www.eng.uci.edu/~kurdahi	USA

Education and Employment Positions

Degrees:
1981- American University of Beirut, Lebanon, BACHELOR OF ENGINEERING, Electrical Engineering
1982- University of Southern California, Los Angeles, CA, USA MASTER OF SCIENCE Computer Engineering
1987- University of Southern California, Los Angeles, CA, USA, DOCTOR OF PHILOSOPHY Computer Engineering

Positions held:

2017	Associate Dean for Graduate and Professional Studies, Samueli School of Engineering, University of Cal
2012	Director, Center for Embedded & Cyber-physical Systems, University of California, Irvine, CA, USA
1998	Professor, University of California, Irvine, CA, USA
1993 - 1998	Associate Professor, University of California, Irvine, CA, USA
1987 - 1993	Assistant Professor, University of California, Irvine, CA, USA

Honors and Awards

IEEE Fellow, 2005
AAAS Fellow, 2008
Distinguished Alumnus Award, American University of Beirut, 2008
Best Paper award, IEEE Trans. On VLSI, 2001
Best Paper award IEEE/ISQED, 2006
Best Paper Award IEEE/ACM ASP-DAC 2016
Distinguished paper DAC, 1984
Distinguished paper EuroDAC 1992
Distinguished paper ASP- DAC 1998
Distinguished paper ISQED 2007

Research Interests

Computer Aided Design and design methodology of large scale systems.

Five **most recent publications**	**Total n^o of publications: 280+**

Fouda, M.E., Kurdahi, F.J., ElTawil, A., & Neftci, E. (2019). Spiking Neural Networks for Inference and Learning: A Memristor-based Design Perspective. *ArXiv, abs/1909.01771*. Also to appear in In Memristive Devices for Brain-Inspired Computing, Querlioz, D, Spika, S, Sebastian A, and Rajendran, B Eds. Elsevier.

Yantir, H.E.; Guo, W.; Eltawil, A.M.; Kurdahi, F.J.; Salama, K.N. An Ultra-Area-Efficient 1024-Point In-Memory FFT Processor. Micromachines 2019, 10, 509.

Minjun Seo and Fadi Kurdahi. 2019. Efficient Tracing Methodology Using Automata Processor. ACM Trans. Embed. Comput. Syst. 18, 5s, Article 80 (October 2019), 18 pages. DOI:https://doi.org/10.1145/3358200

Ahmed E. Khorshid *, Ibrahim N. Alquaydheb, Fadi Kurdahi, Roger Piqueras Jover, Ahmed Eltawil. Biometric Identity Based On Intra-Body Communication Channel Characteristics and Machine Learning. *Sensors* 2020, 20, 1421. doi:10.3390/s20051421

M. Möstl *et al*., "Platform-Centric Self-Awareness as a Key Enabler for Controlling Changes in CPS," in *Proceedings of the IEEE*, vol. 106, no. 9, pp. 1543-1567, Sept. 2018.

Five most relevant publications to this proposal (if different from above)

M. Fouda, S. Lee, J. Lee, A. M. Eltawil and F. Kurdahi, "Mask Technique for Fast and Efficient Training of Binary Resistive Crossbar Arrays," in *IEEE Transactions on Nanotechnology*. Vol. 18, issue 1, pp 704-716, Dec. 2019. doi: 10.1109/TNANO.2019.2927493

M. E. Fouda, E. Neftci, A. Eltawil and F. Kurdahi, "Independent Component Analysis Using RRAMs," in *IEEE Transactions on Nanotechnology*, vol. 18, pp. 611-615, 2019.
doi: 10.1109/TNANO.2018.2880734.

H. E. Yantir, A. M. Eltawil and F. J. Kurdahi, "A Two-Dimensional Associative Processor," in IEEE Transactions on Very Large Scale Integration (VLSI) Systems. DOI: 10.1109/TVLSI.2018.2827262

M. E. Fouda, A. M. Eltawil and F. Kurdahi, "Modeling and Analysis of Passive Switching Crossbar Arrays," in IEEE Transactions on Circuits and Systems I: Regular Papers, vol. 65, no. 1, pp. 270-282, Jan. 2018. doi: 10.1109/TCSI.2017.2714101

M. A. Zidan, A. M. Eltawil, F. Kurdahi, H. A. H. Fahmy and K. N. Salama, "Memristor Multiport Readout: A Closed-Form Solution for Sneak Paths," in *IEEE Transactions on Nanotechnology*, vol. 13, no. 2, pp. 274-282, March 2014.