ASCEND: A Scalable and Energy-Efficient Deep Neural Network Accelerator With Photonic Interconnects

Yuan Li, *Student Member, IEEE*, Ke Wang, *Student Member, IEEE*, Hao Zheng, *Student Member, IEEE*, Ahmed Louri, *Fellow, IEEE*, and Avinash Karanth, *Senior Member, IEEE*

***Abstract*— The complexity and size of recent deep neural network (DNN) models have increased significantly in pursuit of high inference accuracy. Chiplet-based accelerator is considered a viable scaling approach to provide substantial computation capability and on-chip memory for efficient process of such DNN models. However, communication using metallic interconnects in prior chiplet-based accelerators poses a major challenge to system performance, energy efficiency, and scalability. Photonic interconnects can adequately support communication across chiplets due to features such as distance-independent latency, high bandwidth density, and high energy efficiency. Furthermore, the salient ease of broadcast property makes photonic interconnects suitable for DNN inference which often incurs prevalent broadcast communication. In this paper, we propose a scalable chiplet-based DNN accelerator with photonic interconnects named ASCEND. ASCEND introduces (1) a novel photonic network that supports seamless intra- and inter-chiplet broadcast communication, and flexible mapping of diverse convolution layers, and (2) a tailored dataflow that exploits the ease of broadcast property and maximizes parallelism by simultaneously processing computations with shared input data. Simulation results using multiple DNN models show that ASCEND achieves 71% and 67% reduction in execution time and energy consumption, respectively, as compared to other state-of-the-art chiplet-based DNN accelerators with metallic or photonic interconnects.**

***Index Terms*— Deep neural network, photonic interconnect, chiplet, accelerator, dataflow.**

I. INTRODUCTION

RECENT deep neural network (DNN) models have significantly increased in complexity and size with the goal of improving inference accuracy [1]–[8]. As a result, the underlying computing systems must scale up in computation capability and on-chip memory for efficient process of such DNN models [4]. Chiplet-based accelerator [9]–[12] is considered a viable scaling approach as the scaling of a monolithic chip slows down due to concerns related to power density, yield, and fabrication cost [11], [13]. However, communication across chiplets using metallic interconnects in prior chiplet-based accelerators [12] poses a major challenge to system performance, energy efficiency, and scalability. This is because the long-distance communication across chiplets accentuates latency and latency discrepancy, inevitably leading to performance degradation and difficulty in data movement orchestration. Besides, the energy consumption of communication across chiplets is higher than within a monolithic chip [12], [14].

Disruptive technologies such as photonic interconnects can potentially overcome the fundamental limitations of metallic interconnects [15]–[18]. Data can propagate through waveguide within one hop regardless of the distance between source and destination, maintaining low and uniform communication latency in a chiplet-based accelerator. Communication bandwidth can be increased through techniques such as wavelength-division multiplexing (WDM) and space-division multiplexing (SDM) [19]. Photonic interconnects have also shown advantage in energy efficiency for long-distance communication as often seen in chiplet-based accelerators [15], [17]. Despite the above superior features of photonic interconnects, the salient ease of broadcast property [15], [16] makes photonic interconnects especially suitable for DNN inference which often incurs prevalent broadcast communication [20]–[23].

Prior photonic networks [24]–[35] often target inter-processor communication typically observed in CPUs or GPUs, and support uniform bandwidth provision at relatively high cost. Besides, the ease of broadcast property of photonic interconnects is not fully exploited. Some prior photonic networks [27], [34] only employ broadcast to facilitate cache coherence protocol. Other photonic networks [29], [31], [33], though constructed by single-write-multiple-read (SWMR) channels, disable the broadcast capability. As a result, a novel photonic network which is tailored to DNN inference and efficiently supports broadcast communication is necessary.

Manuscript received September 16, 2021; revised March 2, 2022; accepted April 12, 2022. Date of publication May 6, 2022; date of current version June 29, 2022. This work was supported in part by the National Science Foundation under Grant CCF-1702980, Grant CCF-1812495, Grant CCF-1901165, Grant CCF-1953980, Grant CCF-1513606, Grant CCF-1703013, and Grant CCF-1901192. This article was recommended by Associate Editor B. Wu. *(Corresponding author: Yuan Li.)*

Yuan Li, Ke Wang, Hao Zheng, and Ahmed Louri are with the Department of Electrical and Computer Engineering, George Washington University, Washington, DC 20052 USA (e-mail: liyuan5859@gwu.edu; cory@gwu.edu; haozheng@gwu.edu; louri@gwu.edu).

Avinash Karanth is with the School of Electrical Engineering and Computer Science, Ohio University, Athens, OH 45701 USA (e-mail: karanth@ohio.edu).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TCSI.2022.3169953.

Digital Object Identifier 10.1109/TCSI.2022.3169953

Employing photonic interconnects in chiplet-based accelerators also alters the primary target of dataflow optimization. Prior dataflow optimizations for accelerators with metallic interconnects [12], [36]–[41] often prioritize exploiting locality over broadcast communication. For example, some dataflow optimizations [12], [42] exploit locality of weights at the cost of only being able to broadcast input features. By contrast, [40] exploits locality of input features at the cost of only being able to broadcast weights. Due to the distance-independent latency feature and ease of broadcast property of photonic interconnects, a tailored dataflow that enables broadcast of both types of input data (weights and input features) is beneficial, when a photonic network is implemented in a chiplet-based accelerator.

In this paper, we propose a chiplet-based DNN accelerator with photonic interconnects named ASCEND. ASCEND includes (1) a novel photonic network that facilitates massive broadcast communication, and (2) a tailored dataflow that exploits the ease of broadcast property to improve parallelism. The contributions of this paper include:

A Novel Photonic Network: We construct a unit 2D processing element (PE) array by selectively grouping local PEs and corresponding PEs across different chiplets in columns and rows, respectively. A waveguide facilitates the broadcast communication from the global buffer (GLB) to this PE array through WDM while a second waveguide reuses the wavelengths for unicast communication from each individual PE to the GLB. A chiplet-based accelerator is constructed by aggregating multiple such PE arrays and connecting them to the GLB through SDM. The resulting photonic network supports (1) seamless one-hop intra- and inter- chiplet broadcast communication, and (2) flexible mapping of diverse convolution layers at the granularity of a unit 2D PE array.

A Tailored Dataflow: We introduce a broadcast-based output-stationary dataflow that exploits the broadcast communication capability of the proposed photonic network and facilitates high parallelism. Specifically, this dataflow enforces intra-chiplet broadcast of input features and inter-chiplet broadcast of weights by spatially mapping computations with shared input features and weights to columns and rows of PEs in the unit 2D PE arrays, respectively. Furthermore, the output-stationary nature of this dataflow minimizes the unicast communication of writing back intermediate data from PEs to the GLB.

Evaluation and Design Space Exploration: We compare ASCEND with other state-of-the-art chiplet-based accelerators with metallic or photonic interconnects using multiple DNN models. Simulation results show that ASCEND achieves up to 71% and 67% reduction in execution time and energy consumption, respectively. We further perform design space exploration by varying multiple factors such as the size of the unit 2D PE array and the capacity of the GLB.

II. BACKGROUND AND MOTIVATION

A. Communications in DNN

The computations involved in a typical convolution layer can be presented as a 6-dimension nested loop over weight kernels, input feature maps (ifmaps), and output feature maps

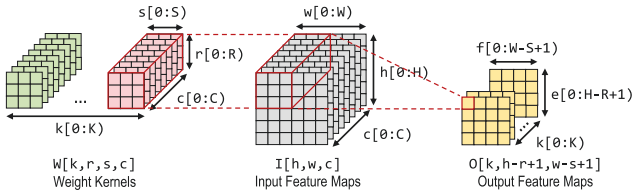

Fig. 1. Computations in a convolution layer.

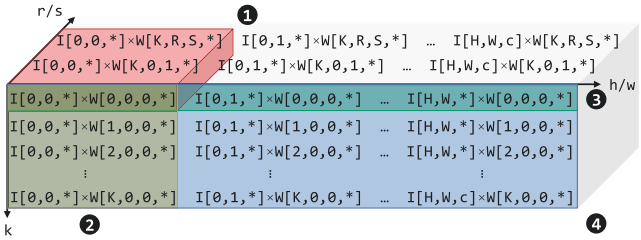

Fig. 2. Multiplications with shared weights or input features.

(ofmaps), assuming the batch size of ifmaps to be 1. As illustrated in Fig. 1 and Algorithm 1, the dimensions include the number of weight kernels (`k`), the number of input channels (`c`), the height (`r`) and width (`s`) of weight kernels, and the height (`h`) and width (`w`) of ifmaps. The height (`e`) and width (`f`) of ofmaps are not independent and can be derived from the above 6 dimensions. In the single-batch case, `e=h-r+1` and `f=w-s+1` (assume stride of 1). As shown in Algorithm 1, there are two types of read-only input data: weights `W[k,r,s,c]` and input features `I[h,w,c]`. Meanwhile, the read-and-write intermediate computation results, known as partial sums (psums), are accumulated to obtain the final output features `O[k,h-r+1,w-s+1]`.

Unlike the dynamic communication patterns often observed in generic applications in CPUs and GPUs, the communication patterns incurred in DNN inference are predetermined by factors such as the dimension values (`C`, `K`, `H`, `W`, `R`, `S`) in the nested loop, the parameters of the underlying computing hardware, and the utilized dataflow. Since each psum `I[h,w,c]×W[k,r,s,c]`is unique and only involved in accumulation once, we focus on identifying the broadcast communication incurred during the separate transmission of two input data types: weights `W[k,r,s,c]` and input features `I[h,w,c]`. Fig. 2 lists multiplications involved in a convolution layer along `k`, `h/w`, and `r/s` dimensions. Please note that the `c` dimension is not shown in Fig. 2, as there is no data sharing and broadcast communication along this dimension. We utilize a symbol "$*$" to represent the value in `c` dimension. We observe that multiplications along the `k` and `r/s` dimensions share the same input feature `I[h,w,c]`, while multiplications along the `h/w` dimension share the same weight `W[k,r,s,c]`, indicating the possible broadcast communication for both types of input data. However, prior dataflow optimizations [12], [40], [42] are not developed to fully exploit the broadcast communication. For example, [42] and [12] spatially distribute multiplications along the `r/s` and `k` dimensions (❶ and ❷ in Fig. 2), respectively, to exploit the locality of weights at the cost of only being able to broadcast input features. By contrast, [40] spatially distributes multiplications along the `h/w` dimension (❸ in Fig. 2) to

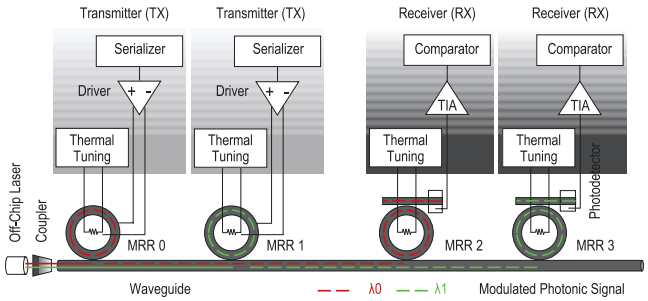

Fig. 3. A WDM photonic link connecting two transmitters and receivers.

Algorithm 1: Nested Loop Representation

```
for c ← [0:C) do
  for k ← [0:K) do
    for h ← [0:H) do
      for w ← [0:W) do
        for r ← [0:R) do
          for s ← [0:S) do
            O[k,h-r+1,w-s+1]+=I[h,w,c]×W[k,r,s,c]
```

exploit the locality of output features at the cost of only being able to broadcast weights. Given that the transmission of both weights and input features can incur broadcast communication, a tailored dataflow that spatially distributes multiplications along the k and h/w dimensions (❹ in Fig. 2) and enables simultaneous broadcast of weights and input features is beneficial when photonic interconnects are employed. Fully-connected layers can also be mathematically framed using the nested loop representation in Algorithm 1, by restricting H=R and W=S. DNN models also include other layer types such as pooling and normalization. Our work focuses on the convolution and fully-connected layers as they dominate the computation and memory communication [43], [44].

B. Photonic Interconnects

We present a photonic link that connects two sets of transmitter and receiver by multiplexing two wavelengths in Fig. 3. The light of wavelengths $\lambda 0$ and $\lambda 1$ is generated by an off-chip laser source and coupled into a waveguide using an optical coupler [45]. Two micro-ring resonators (MRRs) [46], MRR0 and MRR1, work as modulators to modulate the incoming electrical signals on wavelengths $\lambda 0$ and $\lambda 1$, respectively. Another two MRRs, MRR2 and MRR3, work as filters to select modulated wavelengths and forward them to the corresponding photodetectors [16]. Each set of modulator and filter MRRs can only work on a specific wavelength (e.g., MRR0 and MRR2 work only on wavelength $\lambda 0$). The electrical signals generated from photodetectors are then amplified through transimpedance amplifiers (TIAs) and forwarded to comparators to retrieve the initial data being transmitted. All MRRs that function as either modulators or filters, are tuned by separate resistive heaters with specific thermal tuning modules to mitigate thermal and process variations [16], [29]. The example in Fig. 3 only shows the multiplexing of two wavelengths, prior work has shown as many as 64 wavelengths multiplexed in a waveguide with each wavelength operating at 10 Gbps data rate [25], [47]–[49].

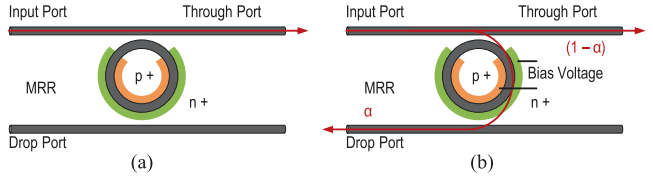

Fig. 4. Optical tunable splitter that works at (a) off-resonance state and (b) a transient state with a split ratio of $\alpha/(1-\alpha)$.

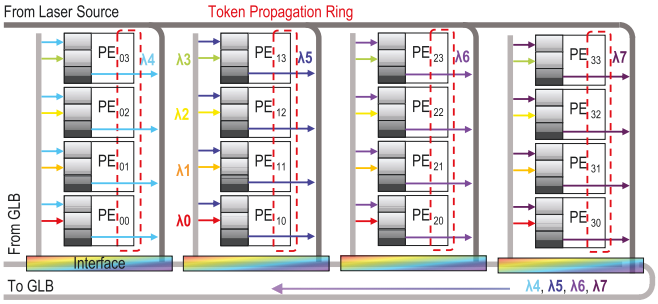

Fig. 5. Architecture and wavelength allocation of a 4×4 unit 2D PE array.

In addition to the common components shown in Fig. 3, ASCEND includes a special component named tunable splitter [50] to facilitate broadcast communication. Different from modulators and filters that work at either on- or off- resonance, a tunable splitter works at a transient state between the on- and off- resonance. As shown in Fig. 4, the regions outside and inside a tunable splitter ring are with n-type and p-type dope, respectively, to form a PIN diode structure. When no bias voltage is applied to the PIN diode as shown in Fig. 4 (a), the tunable splitter is at off-resonance and light from the input port is directly forwarded to the through port. When applying a proper bias voltage to the PIN diode as shown in Fig. 4 (b), the tunable splitter works at the transient state to guide α fraction of light to the drop port while forwarding the remaining $(1-\alpha)$ fraction of light to the through port. The split ratio is defined as $\alpha/(1-\alpha)$. [50] reports that different split ratios in the range of [0.4,1.8] can be obtained by tuning the bias voltage in the range of [0,5V]. The applied bias voltage is tuned by a digital-to-analog converter (DAC). In the case that a split ratio beyond the range of [0.4,1.8] is needed, multiple tunable splitters must be cascaded [51].

Many prior photonic networks [14], [24], [25], [30], [31], [33] for chiplet-based architectures are developed for inter-processor communication typically observed in CPUs and GPUs. The resulting uniform bandwidth provision approach leads to excessive energy and area overhead. For example, the number of MRRs in photonic crossbars in [30], [31], [33] scales quadratically with the number of chiplets. Furthermore, though constructed by SWMR channels which are naturally suitable for broadcast communication, the broadcast capability of the above photonic crossbars are disabled due to power and other concerns. Unlike prior photonic networks for chiplet-based architectures, ASCEND photonic network is tailored for DNN inference and facilitates massive broadcast communication and high parallelism.

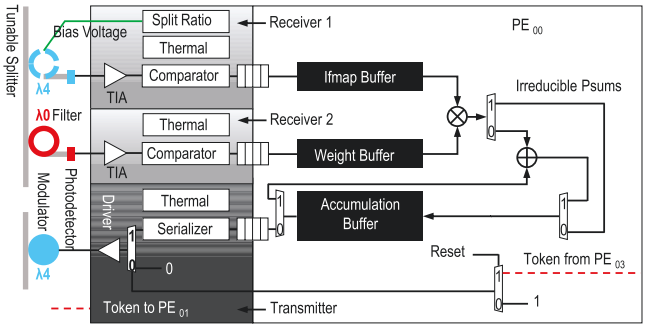

Fig. 6. ASCEND PE architecture, taken PE00 in Fig. 5 as an example.

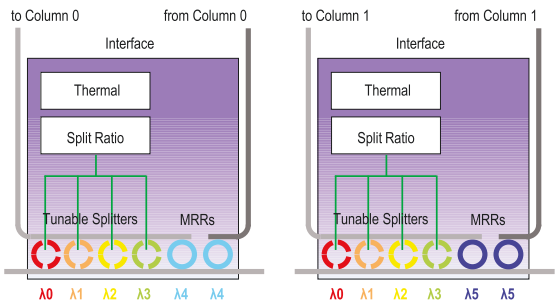

Fig. 7. The interfaces attached to Column0 and Column1 in Fig. 5.

III. ASCEND ARCHITECTURE

A. Unit 2D Processing Element Array

Recall Fig. 2 where we spatially distribute multiplications along both k and h/w dimensions to achieve simultaneous broadcast of input features and weights, respectively. As a result, PEs in a chiplet-based accelerator are grouped into a unit 2D array to accommodate the above multiplications. The purpose of constructing a unit 2D PE array is to (1) explore the optimal organization of PEs with high energy-efficiency, and (2) construct large-scale chiplet-based accelerators in a scalable manner by aggregating one or multiple unit 2D PE arrays.

1) Unit 2D PE Array Architecture: Fig. 5 illustrates the architecture and wavelength allocation of a 4×4 unit 2D PE array. The architectural details of PE00 in Fig. 5 are presented in Fig. 6. Each PE includes a multiply-accumulate (MAC) unit and register buffers to store weights, input features and intermediate psums. There are one transmitter for PE-to-GLB unicast communication and two receivers for GLB-to-PE broadcast communication. Please note that one receiver is connected to a tunable splitter for per-column broadcast communication as the wavelength ($\lambda 4$ in Fig. 6) is shared by all PEs in the same column and only a fraction of light is guided to the photodetector. By contrast, the other receiver is connected to a filter for per-row broadcast communication as the wavelength ($\lambda 0$ in Fig. 6) is dedicated for communication to a PE (PE00 in Fig. 6). Since PEs in a column utilize the same wavelength for PE-to-GLB unicast communication, a token-based approach is implemented for arbitration. As shown in Fig. 5 and Fig. 6, a token is propagated among PEs in a column through a token propagation ring. Interfaces attached to different columns are very similar as shown in Fig. 7. A set of tunable splitters are responsible for guiding an appropriate fraction of light of wavelengths ($\lambda 0$, $\lambda 1$, $\lambda 2$, $\lambda 3$ in Fig. 7) to the corresponding column while forwarding the remaining fraction of light to downstream columns. The split ratio shared by a set of tunable splitters is determined according to the position of the corresponding column. For example, the split ratio values for the interfaces associated with Column0 and Column1 are 1/3 and 1/2, as there are three and two downstream columns, respectively. Within each interface, there are also two MRRs that keep working at on-resonance state to filter and merge the wavelength ($\lambda 4$ for Column0) for per-column broadcast communication and PE-to-GLB unicast communication.

2) Wavelength Allocation: Four wavelengths $\lambda 0$, $\lambda 1$, $\lambda 2$, $\lambda 3$ are utilized to broadcast weights from the GLB to each row of PEs. For example, wavelength $\lambda 0$ is utilized to broadcast weights from the GLB to PE00, PE10, PE20, and PE30 in the first row of the unit 2D PE array. Additional four wavelengths $\lambda 4$, $\lambda 5$, $\lambda 6$, $\lambda 7$ are utilized to broadcast input features from the GLB to each column of PEs. For example, wavelength $\lambda 4$ is utilized to broadcast input features from the GLB to PE00, PE01, PE02, and PE03. The wavelengths for per-column broadcast communication are also reused for PE-to-GLB unicast communication (e.g., wavelength $\lambda 4$ is reused for unicast communication from PEs in the first column to the GLB). Please note that multiple independent waveguides can be implemented using SDM to increase the bandwidth provision for PE-to-GLB communication. All eight wavelengths involved in Fig. 4, $\lambda 0$, $\lambda 1$, $\lambda 2$, $\lambda 3$, $\lambda 4$, $\lambda 5$, $\lambda 6$, $\lambda 7$, are multiplexed in a waveguide using WDM.

3) Network Power Consumption of Unit 2D PE Array: The network power consumption of a unit 2D PE array is directly affected by its size (number of PEs involved) and shape (ratio of array height and width). As array size increases, the overall power consumption of modulators and associated heaters decreases as each transmitter can broadcast to an increasing volume of receivers. However, the laser power consumption increases drastically due to insertion loss increase when more PEs are attached to each broadcast channel. We explore the impact of array size on network power consumption and observe that optimal power consumption is obtained at 16×16 array size. For simplicity, we continue using the 4×4 unit 2D PE array to explain the proposed ASCEND architecture. Similarly, the network power consumption of a unit 2D PE array is also affected by the shape of the array, given a fixed number of PEs involved. Non-square array shapes (e.g., 2×8 and 8×2) inevitably lead to insertion loss imbalance between per-column and per-row broadcast channels. As we assume that each wavelength is generated with similar power from the off-chip laser source, a fraction of power of wavelengths utilized in broadcast channels with low insertion loss will be wasted.

B. ASCEND Network

1) Network Overview: Fig. 8 presents an ASCEND architecture with eight accelerator chiplets and eight PEs per accelerator chiplet. This chiplet-based accelerator is constructed by aggregating four 4×4 unit 2D PE arrays. Given the per-column

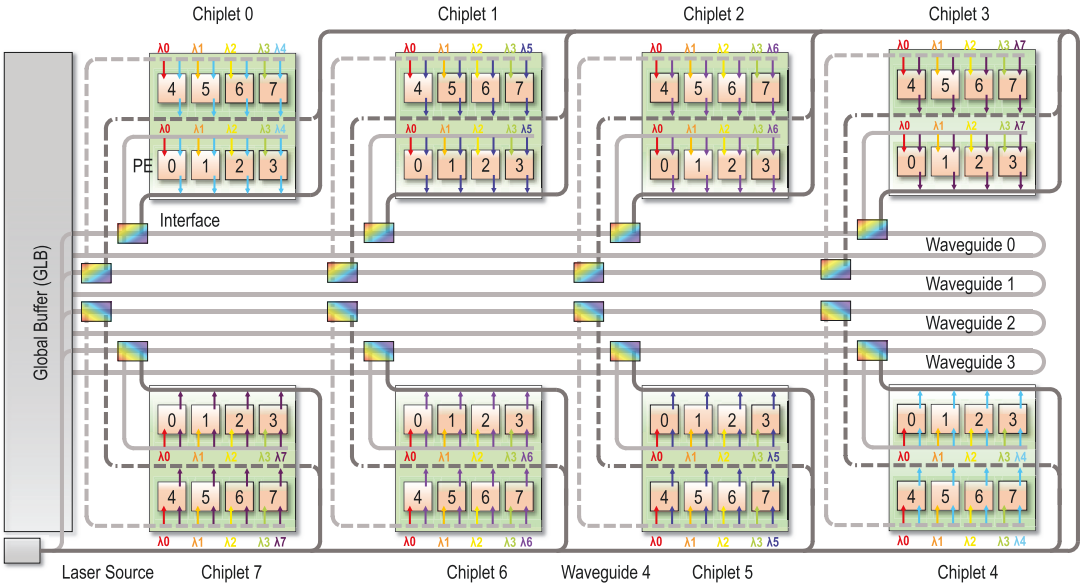

Fig. 8. ASCEND architecture with eight accelerator chiplets and eight PEs per accelerator chiplet, constructed by four 4 × 4 unit 2D PE arrays. The four unit 2D PE arrays are connected to the GLB through four separate waveguides (`Waveguide0`, `Waveguide1`, `Waveguide2`, and `Waveguide3`) using SDM. Another waveguide (`Waveguide4`) is utilized to support simultaneous GLB-to-PE broadcast and PE-to-GLB unicast communication.

broadcast communication support of a unit 2D PE array discussed before, we allocate a column of PEs in a unit 2D PE array to a single accelerator chiplet (e.g., `PE0`, `PE1`, `PE2`, and `PE3` in `Chiplet0` are from the same column of a unit 2D PE array). Therefore, the per-column broadcast communication in a unit 2D PE array is equivalent to intra-chiplet broadcast communication in the constructed chiplet-based accelerator. Similarly, we allocate a row of PEs in a unit 2D PE array to the same position of different accelerator chiplets (e.g., `PE0` in `Chiplet0`, `PE0` in `Chiplet1`, `PE0` in `Chiplet2`, and `PE0` in `Chiplet3` are from the same row of a unit 2D PE array), making the per-row broadcast communication equivalent to inter-chiplet broadcast communication. In Fig. 8, each row of sixteen PEs across four accelerator chiplets belong to a unit 2D PE array. PEs within the same accelerator chiplet but in different unit 2D PE arrays are separately connected by waveguides presented by solid and dashed lines. The four involved unit 2D PE arrays in Fig. 8 are connected to the GLB die with four separate waveguides using SDM. For example, the unit 2D PE array including `PE0`, `PE1`, `PE2`, and `PE3` in `Chiplet0` to `Chiplet3` is connected to the GLB with `Waveguide0`. We observe wavelength reuse between unit 2D PE arrays as separate waveguides are utilized. In the ASCEND architecture shown in Fig. 8, wavelengths $\lambda 0$, $\lambda 1$, $\lambda 2$, and $\lambda 3$ are reused for inter-chiplet broadcast communication while wavelengths $\lambda 4$, $\lambda 5$, $\lambda 6$, and $\lambda 7$ are reused for intra-chiplet broadcast communication. `Waveguide4` is used to deliver light to each PE for PE-to-GLB unicast communication.

2) Inter-Chiplet Broadcast Communication: The inter-chiplet broadcast function in ASCEND broadcasts the same weight to PEs in the same position of different accelerator chiplets. The broadcast communication from the GLB to `PE0` in all eight accelerator chiplets is done by modulating wavelength $\lambda 0$ on both `Waveguide0` and `Waveguide3`. Similarly, the broadcast communication from the GLB to `PE1` in all accelerator chiplets is done by modulating wavelength $\lambda 1$ on both `Waveguide0` and `Waveguide3`, while the broadcast communication from the GLB to `PE4` in all accelerator chiplets is done by modulating wavelength $\lambda 0$ on both `Waveguide1` and `Waveguide2`. During inter-chiplet broadcast communication, the tunable splitters in the interfaces along a waveguide are tuned to appropriate split ratios to guide a fraction of laser power in $\lambda 0$ to $\lambda 3$ to the local accelerator chiplet while forwarding the remaining fraction of laser power to downstream accelerator chiplets. For example, the tunable splitters in the interfaces attached to `Chiplet0` are all tuned to have a split ratio of `1/3` because there are in total 3 downstream chiplets along either `Waveguide0` or `Waveguide1`. The laser power at the drop port of a tunable splitter is collected and forwarded to the PE with a filter working on the same wavelength, which means this particular PE is a destination of inter-chiplet broadcast communication.

Algorithm 2: ASCEND Dataflow

```
// Package level
for e1 ← [0:E1) do
 for f1 ← [0:F1) do
  parallel_for e2 ← [0:E2) do
   parallel_for f2 ← [0:F2) do
    parallel_for k1 ← [0:K1) do
// Chiplet level
for k2 ← [0:K2) do
 parallel_for k3 ← [0:K3) do
  parallel_for e3 ← [0:E3) do
   parallel_for f3 ← [0:F3) do
// PE level
for c ← [0:C) do
 for r ← [0:R) do
  for s ← [0:S) do
   k=k3+K3×(k2+K2×k1)
   e=e3+E3×(e2+E2×e1)
   f=f3+F3×(f2+F2×f1)
   O[k,e,f]+=I[r+e-1,s+f-1,c]×W[k,r,s,c]
```

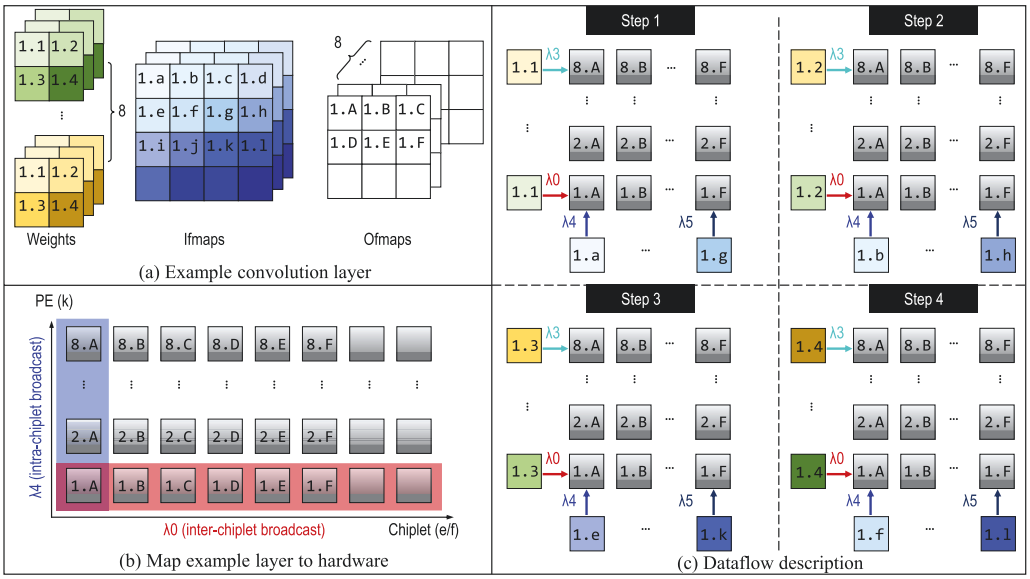

Fig. 9. Processing a convolution layer `[r,s,e,f,c,k]=[2,2,4,4,3,8]` on the ASCEND architecture as shown in Fig. 8 that supports intra- and inter-chiplet broadcast communication. ASCEND dataflow processes output features on the same `e/f` dimension on different accelerator chiplets while processing output features with different `k` dimension values on different PEs in the same accelerator chiplet.

3) Intra-Chiplet Broadcast Communication: The intra-chiplet broadcast function in ASCEND broadcasts the same input feature to PEs in the same accelerator chiplet. The broadcast communication from the GLB to all PEs in `Chiplet0` is done by modulating wavelength $\lambda 4$ on both `Waveguide0` and `Waveguide1`. Similarly, the broadcast communication from the GLB to all PEs in `Chiplet1` is done by modulating wavelength $\lambda 5$ on both `Waveguide0` and `Waveguide1`, while the broadcast communication from the GLB to all PEs in `Chiplet4` is done by modulating wavelength $\lambda 4$ on both `Waveguide2` and `Waveguide3`. During intra-chiplet broadcast communication, the MRR filters in the interfaces along a waveguide work at on-resonance state and completely guide wavelengths for intra-chiplet broadcast communication to the drop port. The laser power is then collected and propagated through local PEs. The tunable splitter attached to the receiver of a specific PE is utilized to guide an appropriate fraction of laser power to the corresponding photodetector while forwarding the remaining fraction of laser power to downstream PEs. For example, the tunable splitter attached to `PE0` of `Chiplet0` is tuned to a split ratio of `1/3` as there are three downstream PEs (`PE1`, `PE2`, and `PE3`).

4) PE-to-GLB Unicast Communication: The intra- and inter- chiplet broadcast functions in ASCEND only address the transmission of input data: weights and input features. The intermediate psums and final output features are transmitted to the GLB through PE-to-GLB unicast function. This function reuses the wavelengths originally allocated for intra-chiplet broadcast communication. For example, wavelength $\lambda 4$ is allocated for both intra-chiplet broadcast communication and PE-to-GLB unicast communication in `Chiplet0`. The wavelength conflict of these two functions is resolved by implementing separate waveguides. As local PEs share the same wavelength for PE-to-GLB unicast communication, a token-based approach is employed. PE that possesses the single-bit token can transmit its intermediate psums or output features back to the GLB. Once the transmission is complete, the single-bit token is released and propagated to the next local PE through an electrical token propagation ring. The token is originally possessed by the first local PE after reset (active low reset signal in Fig. 6). Because of the uniform computation operations across all PEs, a single-bit electrical token propagation ring is sufficient compared to more sophisticated token arbitration waveguide approach [34]. The bandwidth for PE-to-GLB unicast communication is smaller than the bandwidth for GLB-to-PE broadcast communication. This bottleneck is alleviated by adopting an output-stationary-based dataflow as discussed in the following section. The bandwidth for PE-to-GLB unicast communication can also be expanded by implementing multiple waveguides using SDM.

C. ASCEND *Dataflow*

ASCEND dataflow, as shown in Algorithm 2 and Fig. 9, is optimized based on three unique features of the proposed photonic network. First, ASCEND supports intra- and inter-chiplet broadcast communication by leveraging the ease of broadcast property of photonic interconnects. Second, by using an output-stationary dataflow, we minimize data exchange between PEs. Third, output-stationary dataflow prioritizes reducing psum movement, which significantly reduces the bandwidth demand for PE-to-GLB unicast communication. Moreover, by multiplexing different wavelengths, we can increase the number of psums sent simultaneously back to GLB from different chiplets.

Consider the convolution layer shown in Fig. 9 (a) as an example. We represent different weight kernels (output channel `k` dimension) with different colors, and label a weight in a specific kernel using `X:Y` terminology where `X` and `Y` represent the input channel in the `C` dimension and the position of this weight, respectively. Input features are represented using the same terminology as for weights. For output features, `X` in the `X:Y` terminology represents the output channel in the

TABLE I
NETWORK PARAMETERS

Simba	Chiplet level	Electrical mesh 20 Gbps / PE read / write bandwidth
	Package level	Electrical mesh 320 Gbps / chiplet read / write bandwidth
POPSTAR	Chiplet level	Electrical mesh 20 Gbps / PE read / write bandwidth
	Package level	Photonic crossbar 310 Gbps / chiplet read bandwidth 100 Gbps / chiplet write bandwidth 10 wavelengths, 10 Gbps / wavelength
ASCEND	Chiplet level	20 Gbps / PE read bandwidth 10 Gbps / PE write bandwidth (shared)
	Package level	340 Gbps / chiplet read bandwidth 20 Gbps / chiplet write bandwidth 32 wavelengths, 10 Gbps / wavelength

TABLE II
STANDARD PHOTONIC PARAMETERS

Component	Value	Component	Value
Laser source	5 dB [47]	Ring drop	1 dB [54]
Coupler	1 dB [47]	Ring through	0.02 dB [55]
Splitter	0.2 dB [49]	Photodetector	0.1 dB [47]
Waveguide	1 dB/cm [47]	Waveguide-to-receiver	0.5 dB [56]
Waveguide bend	1 dB [56]	Receiver sensitivity	-20 dBm [47]
Waveguide crossover	0.05 dB [56]	Ring heating	2 mW [57]

k dimension. Fig. 9 (b) describes how the example convolution layer is mapped to the ASCEND architecture shown in Fig. 8 to fully exploit the broadcast capability of the ASCEND photonic network. We map two rows of output features in an ofmap to different chiplets(E2=2, F2=3 in the dataflow shown in Algorithm 2) while filling the rest PEs in each chiplet with corresponding output features in other ofmaps (K3=8 in the dataflow shown in Algorithm 2). As we allocate output features at the same ofmap to different accelerator chiplets, the inter-chiplet broadcast capability of ASCEND photonic network can be leveraged to transmit weights from the GLB to PEs. Meanwhile, as we allocate output features from different ofmaps to PEs within a chiplet, the intra-chiplet broadcast capability of ASCEND photonic network can be leveraged to transmit input features. By doing so, both types of input data are transmitted to PEs through broadcast communication.

Fig. 9 (c) describes the detailed computation and communication operations involved in one iteration of the c loop in Algorithm 2. Since R=S=2, the operations are done in four steps. We focus on computation and communication operations related to two PEs responsible for operations related to output features 1.A and 1.F. Operations related to other PEs can be easily inferred. In Step1, weight labeled 1.1 and in green color is transmitted to 1.A and 1.F using inter-chiplet broadcast wavelength $\lambda 0$. Meanwhile, input features labeled 1.a and 1.g are transmitted to 1.A and 1.F using intra-chiplet broadcast wavelengths $\lambda 4$ and $\lambda 5$, respectively. 1.A and 1.F perform MAC operations when corresponding weights and input features are delivered, and generate 1.1×1.a and 1.1×1.g, respectively. There are similar operations in the following steps. The psums generated at Step4 are stored in the local accumulation buffers for the next iteration of the c loop (Line13 in Algorithm 2). Once the entire c loop is completed, the final output features are obtained and transmitted to the GLB.

D. Flexible Mapping of Convolution Layers

Consider a layer [r,s,e,f,c,k]=[2,2,2,2,3,16], the number of output features on an ofmap is e×f=2×2=4 while the number of output channels is k=16. When mapping this convolution layer to the ASCEND architecture shown in Fig. 8, we observe that only four accelerator chiplets are utilized. Meanwhile, the computations along k dimension have to be iteratively performed while there are idle accelerator chiplets in the system. To resolve this issue, we virtually construct a 16×4 PE array instead of a 8×8 PE array in the k and e/f dimensions by simultaneously broadcasting the same input feature in Waveguide0 to Waveguide3 in Fig. 8. This approach exploits Line6 of the ASCEND dataflow shown in Algorithm 2.

Consider another convolution layer with parameters [r,s,e,f,c,k]=[2,2,4,4,3,4], the number of output features on an ofmap is e×f=4×4=16 while the number of output channels k=4. This represents an opposite situation as compared to the above example. When mapping this convolution layer to the ASCEND architecture shown in Fig. 8, we observe that only 4 PEs in each accelerator chiplet are utilized. Meanwhile, the computations along e/f dimension have to be iteratively performed while there are idle PEs in all accelerator chiplets in the system. To resolve this issue, we virtually construct a 4 × 16 PE array in the k and e/f dimensions by simultaneously broadcasting the same weight in Waveguide0 to Waveguide3 in Fig. 8. This approach exploits Line10 and Line11 of the ASCEND dataflow shown in Algorithm 2.

IV. EVALUATION METHODOLOGY

A. Simulation Platform

In order to evaluate ASCEND and other chiplet-based DNN accelerators [12], [33], we extend the open-source MAESTRO simulator [52] to support the non-uniform distribution of latency and bandwidth between PEs. The execution time includes both computation time and communication time. The extended simulator tracks the number of arithmetic operations and the number of accesses to each in-package memory hierarchy (e.g. GLB, local register buffer, etc.) to calculate the computation time and in-package communication time, respectively. The calculation takes the hierarchical network architecture into account and ensures that communication does not exceed the bandwidth limit of the corresponding link. The delay for tuning the optical tunable splitters is set to 500 ps [50]. The off-package communication time is obtained from the DRAMSim2 simulator [53]. We assume that the communication time is maximally overlapped by the computation time.

B. Power Model

We evaluate the power consumption of computations using *Synopsys Design Compiler*. The power consumption values

TABLE III
AGGRESSIVE PHOTONIC PARAMETERS

Component	Value	Component	Value
Laser source	5 dB [47]	Ring drop	0.7 dB [55]
Coupler	1 dB [47]	Ring through	0.01 dB [58]
Splitter	0.2 dB [49]	Photodetector	0.1 dB [47]
Waveguide	1 dB/cm [47]	Waveguide-to-receiver	0.5 dB [56]
Waveguide bend	0.01 dB [59]	Receiver sensitivity	-26 dBm [60]
Waveguide crossover	0.05 dB [56]	Ring heating	320 μW [61]

of accessing in-package memory hierarchies and off-package DRAM are obtained using CACTI 6.0 [62] and DRAMSim2, respectively. The power consumption of the in-package metallic-based interconnects is obtained using DSENT [55], while the power consumption of photonic interconnects is derived from Equation (1):

$$P_{total} = P_{laser} + P_{TX} + P_{RX} + P_{thermal} \tag{1}$$

The overall power consumption $\mathtt{P_{total}}$ includes three parts: laser power $\mathtt{P_{laser}}$, power consumption of transmitting circuitry $\mathtt{P_{TX}}$, and power consumption of receiving circuitry $\mathtt{P_{RX}}$. We calculate $\mathtt{P_{TX}}$ and $\mathtt{P_{RX}}$ using the same parameters as in [61], [63]. Please note that the power consumption for ring heating is not included in both $\mathtt{P_{TX}}$ and $\mathtt{P_{RX}}$. The values for $\mathtt{P_{TX}}$ and $\mathtt{P_{RX}}$ are 0.9 mW and 0.6 mW, respectively.

The laser power $\mathtt{P_{laser}}$ includes four parts: photodetector sensitivity $\mathtt{P_{rs}}$, insertion loss $\mathtt{C_{loss}}$, extinction ratio power penalty $\mathtt{P_{extinction}}$, and system margin $\mathtt{M_{system}}$, as shown in Equation (2):

$$P_{laser} = P_{rs} + C_{loss} + P_{extinction} + M_{system} \tag{2}$$

Table II and Table III list standard and aggressive photonic parameters, respectively, from which the photodetector sensitivity $\mathtt{P_{rs}}$ and insertion loss $\mathtt{C_{loss}}$ can be obtained or derived. We adopt the standard photonic parameters in Table II for energy consumption estimation, unless otherwise stated. $\mathtt{P_{extinction}}$ represents the power penalty caused by extinction ratio which is assumed to be 2 dB [64]. System margin $\mathtt{M_{system}}$ is assumed to be 4 dB [65]. The purpose of the system margin is to allocate a certain amount of power to additional sources of power penalty that may develop during the system lifetime.

C. Chiplet-Based DNN Accelerators for Comparison

ASCEND is compared with Simba [12] and POPSTAR [33]. Simba is the state-of-the-art chiplet-based DNN accelerator with only metallic interconnects. To the best of our knowledge, there are no chiplet-based DNN accelerators with photonic interconnects. Hence, we select a chiplet-based architecture POPSTAR originally designed for general applications, and replace the general CPU chiplets with accelerator chiplets in Simba to create another baseline for fair comparison. We assume the system includes 32 chiplets and 32 PEs per chiplet for ASCEND. PE clock frequency is set to 1 GHz similar to [12]. ASCEND adopts 16×16 unit 2D PE array unless otherwise stated. To maintain the same computation capability, the MAC vector width of each PE is 32 in ASCEND. The local buffer size of a PE in ASCEND is 4 kB (128 B per unit MAC vector width) while the local buffer size of a PE in Simba and POPSTAR is 43 kB [12]. The GLB size in ASCEND is 2 MB (64 B per unit MAC vector width), which is the same as in Simba and POPSTAR [12]. The network parameters of ASCEND and two other baselines are listed in Table I. We attempt to keep the bandwidth values at both chiplet and package levels comparable across ASCEND and two baselines. For example, we keep the bandwidth values at chiplet level the same in ASCEND and two baselines by adjusting the clock frequency of the electrical mesh networks in Simba and POPSTAR. However, some bandwidth values cannot be tuned to be exactly the same due to specific features of different network architectures.

D. Benchmarks

We choose four DNN models, VGG-16 [5], ResNet-50 [1], DenseNet-201 [2], and EfficientNet-B7 [8] as the evaluation benchmarks. ResNet-50 includes more variations of weight kernel size and computation intensity, while VGG-16 includes more communication-intensive fully-connected layers that can test network performance in extreme scenarios. There are 21 and 12 different convolution or fully-connected layers in ResNet-50 and VGG-16, respectively. We will test all 33 layers in a layer-by-layer manner, as each layer exhibits different parameters which have implications on performance and energy consumption of our design and the other two baselines. Please note that we have removed redundant layers with the same configuration parameters. For example, `res2a_branch1` in ResNet-50 has been removed because it has the same configuration parameters as `res2[a-c]_branch2c`. Additionally, we accumulate the execution time and energy consumption values of all layers to obtain an implication of the overall execution time and energy consumption in a complete inference pass. Please note that only the convolution and fully-connected layers are taken into account during the accumulation process.

V. EXPERIMENT RESULTS

A. Execution Time and Energy Consumption

Fig. 10 shows the execution time comparison of ASCEND, Simba and POPSTAR in 33 different ResNet-50 and VGG-16 layers. The execution time values are normalized to the execution time of Simba. As compared to Simba, ASCEND achieves execution time reduction in the range of 21% (L1: `conv1`) to 75% (L21: `fc-1000`). The difference in reduction of execution time comes from (1) the average number of hops of inter-chiplet communication in Simba, (2) the ofmap `e/f` dimension and output channel `k` dimension that determine the utilization rate of PEs in ASCEND, and (3) the input feature reuse distance that largely determines the intra-chiplet broadcast efficiency in ASCEND. As compared to POPSTAR, ASCEND achieves execution time reduction in the range of 7% (L8: `res3[a-d]_branch2b`) to 55% (L6: `res3a_branch1`). This indicates the effectiveness of the architecture and dataflow co-design. ASCEND performs better than POPSTAR because (1) ASCEND exploits the ease of broadcast feature better than POPSTAR through package-level

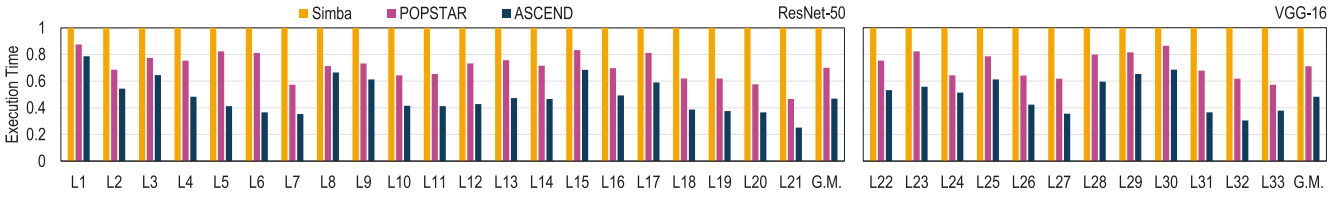

Fig. 10. Per-layer execution time comparison of Simba, POPSTAR and ASCEND. All values are normalized to Simba.

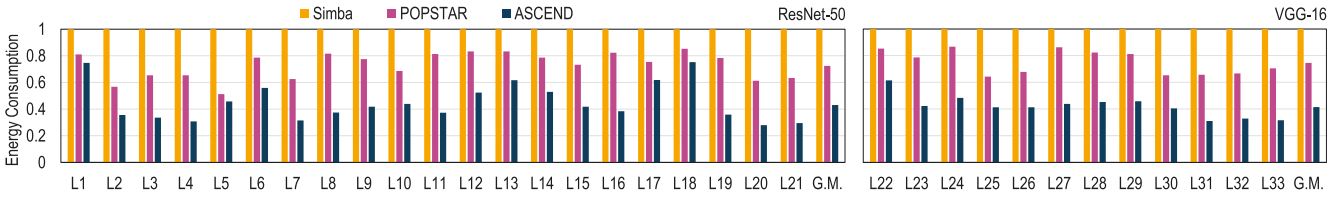

Fig. 11. Per-layer energy consumption comparison of Simba, POPSTAR and ASCEND. All values are normalized to Simba.

data partition, and (2) ASCEND allocates higher bandwidth for communication between the GLB and accelerator chiplets. On average, ASCEND performs 52% and 29% faster than Simba and POPSTAR, respectively.

Fig. 11 shows the energy consumption comparison of ASCEND, Simba and POPSTAR in 33 different ResNet-50 and VGG-16 layers using standard photonic parameters listed in Table II. The energy consumption values are normalized to the energy consumption of Simba. As compared to Simba, ASCEND achieves energy consumption saving in the range of 25% (L1: `conv1`) to 72% (L20: `res5[b-c]_branch2a`). This mainly comes from the low energy consumption of inter-chiplet communication in ASCEND. As compared to POPSTAR, ASCEND achieves energy consumption saving in the range of 7% (L1: `conv1`) to 56% (L33: `fc-1000`). The energy savings observed in different layers are similar because ASCEND photonic inter-chiplet network requires fewer MRRs than the photonic crossbar in POPSTAR. On average, we observe that ASCEND achieves 57% and 46% energy saving as compared to Simba and POPSTAR, respectively.

In addition to the layer-by-layer estimation, we compare the execution time of a complete inference in four different DNN models in Simba, POPSTAR and ASCEND. The results are shown in Fig. 12 (a). We observe that ASCEND achieves execution time reduction in the range of 47% (DenseNet) and 71% (ResNet-50) as compared to Simba. We adopt both standard photonic parameters listed in Table II and aggressive photonic parameters listed in Table III for energy consumption estimation of POPSTAR and ASCEND and present the results in Fig. 12 (b). When using the standard photonic parameters, ASCEND achieves energy reduction in the range of 37% (DenseNet) and 67% (ResNet-50) as compared to Simba. When more aggressive photonic parameters are utilized, more energy reduction in the range of 47% (DenseNet) and 74% (ResNet-50) is achieved by ASCEND as compared to Simba.

B. Analysis on a ResNet-50 Inference Pass

We present the detailed analysis of execution time and energy consumption (using standard photonic parameters listed in Table II) of a complete ResNet-50 inference. Please note that the values only include the convolution layers and

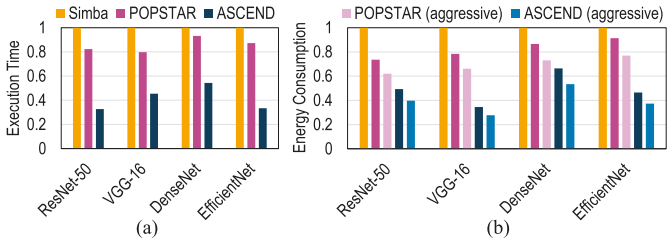

Fig. 12. Normalized (a) execution time and (b) energy consumption of one complete inference in different DNN models.

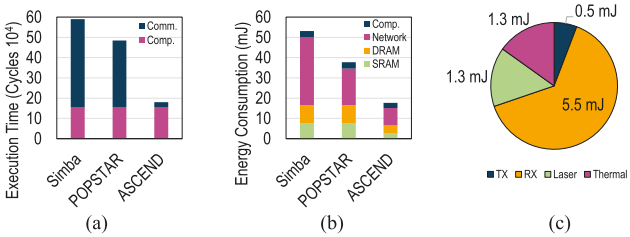

Fig. 13. (a) Execution time and (b) energy consumption breakdown of one complete ResNet-50 inference when comparing ASCEND with Simba and POPSTAR. (c) Network energy consumption breakdown of ASCEND in one complete ResNet-50 inference.

fully-connected layers. We make several observations from the execution time diagram in Fig. 13 (a). First, the numbers of cycles for computation are the same in Simba and POPSTAR, as these two baselines have the same chiplet architecture and dataflow. Second, the numbers of cycles for communication in Simba and POPSTAR are higher than the number of cycles for computation, taking 73% and 66% of overall execution time. Third, the number of cycles for communication in ASCEND is very small due to direct connection between the GLB and each PE, and fully leveraging the broadcast capability of photonic interconnects. Fig. 13 (b) illustrates the energy consumption breakdown of Simba, POPSTAR and ASCEND when processing a complete ResNet-50 inference. The energy reduction as compared to Simba and POPSTAR mainly comes from (1) lower energy consumption of communication network and (2) fewer accesses to the memory hierarchy. When breaking down the network energy consumption of ASCEND as shown in Fig. 13 (c), we observe that the energy consumption values for thermal heating, laser, transmitters and receivers

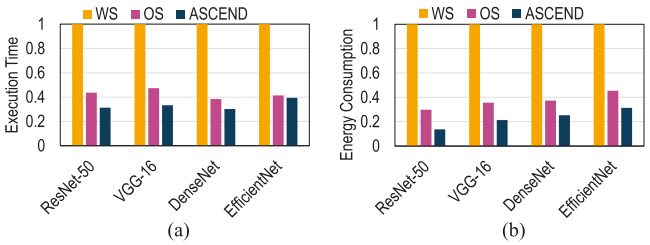

Fig. 14. (a) Execution time and (b) energy consumption comparison when applying weight-stationary [12], output-stationary [40], and ASCEND dataflows to the ASCEND architecture. All values are normalized to weight-stationary dataflow.

are 1.3 mJ, 1.3 mJ, 0.5 mJ, and 5.5 mJ, respectively. The significant difference of energy consumption values of P_{TX} and P_{RX} illustrates that our design successfully leverages the broadcast capability of photonic interconnects. The ASCEND throughput and energy consumption are 5649 frames per second and 21.7 mJ when running ResNet-50 model and assuming batch size of one.

C. Impact of ASCEND Dataflow

Fig. 14 (a) shows the execution time comparison of weight-stationary dataflow in [12], output-stationary dataflow in [40] with partition along k dimension at the package level, and ASCEND dataflow. All three dataflows are implemented on the ASCEND architecture for fair comparison. The weight-stationary dataflow does not fully exploit the two-level broadcast capability of ASCEND photonic network. Partitions along k dimension at package level and along c dimension at chiplet level prevent full utilization of inter-chiplet weight broadcast and intra-chiplet input feature broadcast, respectively. Further, the weight-stationary dataflow incurs inter-PE communication which yields high latency overhead in ASCEND photonic network. The average execution time reduction of ASCEND dataflow over weight-stationary dataflow is 65%. The output-stationary dataflow [40] is originally designed for single-chip DNN accelerators. We extend it by partitioning along k dimension at the package level. The output-stationary dataflow [40] maps output features to PEs one ofmap at a time. Due to mismatch between ofmap size and system scale, full broadcast capability of ASCEND photonic network can not be often achieved. The average execution time reduction of ASCEND dataflow over the output-stationary dataflow is 23%.

Fig. 14 (b) shows the energy consumption comparison of all three dataflows implemented on the ASCEND architecture using the standard photonic parameters listed in Table II. The average energy saving of ASCEND dataflow over the weight-stationary dataflow [12] is 84%. The excessive energy consumed by the weight-stationary dataflow mainly comes from (1) excessive GLB access due to prevalent inter-PE communication, and (2) high fraction of unicast communication that leads to high modulation energy. The average energy saving of ASCEND dataflow over the output-stationary dataflow [40] is 39%.

D. Area Estimation

We estimate the area of ASCEND PE (excluding the transmitter and receivers) using Synopsys Design Compiler and a 28 nm technology library. The area of PE excluding the transmitter and receivers is 0.72 mm^2. We assume that the area for a transmitter or a receiver is 0.0096 mm^2 per wavelength [66]. Hence, the area overhead of the peripheral circuity (E/O and O/E) of an ASCEND PE is about 3.9%. The area of an accelerator chiplet in ASCEND is 24.07 mm^2. When assuming 5 μm MRR radius [67], the overall area of MRRs is 0.01 mm^2. When further assuming 4 electrical wires (for data transmission and thermal tuning) per MRR and 36 μm micro-bump pitch size [68], the overall area of micro-bumps is 0.68 mm^2. As most MRRs and micro-bumps can be implemented underneath the accelerator chiplet, we assume that they do not incur extra area overhead.

VI. Conclusion

In this paper, we propose a chiplet-based DNN accelerator with photonic interconnects named ASCEND. The salient features of ASCEND include (1) a novel photonic network that supports seamless intra- and inter- chiplet broadcast communication and flexible mapping of diverse convolution layers, and (2) a tailored dataflow that exploits the ease of broadcast property of photonic interconnects to maximize parallelism in DNN inference. The combined benefits of the above two features provide high-performance and energy-efficient communication support for scalable chiplet-based DNN accelerators. Simulation studies using multiple DNN models show that ASCEND achieves significant reduction in execution time and energy consumption, and exhibits better scalability, as compared to other state-of-the-art chiplet-based accelerators.

References

[1] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.

[2] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 4700–4708.

[3] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2012, pp. 1097–1105.

[4] R. Mayer and H.-A. Jacobsen, "Scalable deep learning on distributed infrastructures: Challenges, techniques, and tools," *ACM Comput. Surv.*, vol. 53, no. 1, pp. 1–37, Jan. 2021.

[5] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014, *arXiv:1409.1556*.

[6] V. Sze, Y.-H. Chen, T.-J. Yang, and J. S. Emer, "Efficient processing of deep neural networks: A tutorial and survey," *Proc. IEEE*, vol. 105, no. 12, pp. 2295–2329, Dec. 2017.

[7] C. Szegedy *et al.*, "Going deeper with convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 1–9.

[8] M. Tan and Q. Le, "EfficientNet: Rethinking model scaling for convolutional neural networks," in *Proc. Int. Conf. Mach. Learn. (ICML)*, 2019, pp. 6105–6114.

[9] G. Ascia, V. Catania, A. Mineo, S. Monteleone, M. Palesi, and D. Patti, "Improving inference latency and energy of DNNs through wireless enabled multi-chip-module-based architectures and model parameters compression," in *Proc. 14th IEEE/ACM Int. Symp. Netw.-on-Chip (NOCS)*, Sep. 2020, pp. 1–6.

[10] R. Hwang, T. Kim, Y. Kwon, and M. Rhu, "Centaur: A chiplet-based, hybrid sparse-dense accelerator for personalized recommendations," in *Proc. ACM/IEEE 47th Annu. Int. Symp. Comput. Archit. (ISCA)*, May 2020, pp. 968–981.

[11] A. Kannan, N. E. Jerger, and G. H. Loh, "Enabling interposer-based disintegration of multi-core processors," in *Proc. 48th Int. Symp. Microarchitecture*, Dec. 2015, pp. 546–558.

[12] Y. S. Shao *et al.*, "Simba: Scaling deep-learning inference with multi-chip-module-based architecture," in *Proc. 52nd Annu. IEEE/ACM Int. Symp. Microarchitecture*, Oct. 2019, pp. 14–27.

[13] X. Hu, D. Stow, and Y. Xie, "Die stacking is happening," *IEEE Micro*, vol. 38, no. 1, pp. 22–28, Jan. 2018.

[14] P. Fotouhi, S. Werner, J. Lowe-Power, and S. J. B. Yoo, "Enabling scalable chiplet-based uniform memory architectures with silicon photonics," in *Proc. Int. Symp. Memory Syst.*, Sep. 2019, pp. 222–234.

[15] D. A. B. Miller, "Rationale and challenges for optical interconnects to electronic chips," *Proc. IEEE*, vol. 88, no. 6, pp. 728–749, Jun. 2000.

[16] D. A. B. Miller, "Device requirements for optical interconnects to silicon chips," *Proc. IEEE*, vol. 97, no. 7, pp. 1166–1185, Jul. 2009.

[17] A. Shacham, K. Bergman, and L. P. Carloni, "On the design of a photonic network-on-chip," in *Proc. 1st Int. Symp. Netw.-on-Chip (NOCS)*, May 2007, pp. 53–64.

[18] R. Soref, "The past, present, and future of silicon photonics," *IEEE J. Sel. Topics Quantum Electron.*, vol. 12, no. 6, pp. 1678–1687, Nov. 2006.

[19] K. Bergman, L. P. Carloni, A. Biberman, J. Chan, and G. Hendry, *Photonic Network-on-Chip Design*. New York, NY, USA: Springer, 2014.

[20] H. Kwon, A. Samajdar, and T. Krishna, "MAERI: Enabling flexible dataflow mapping over DNN accelerators via reconfigurable interconnects," in *Proc. ACM Int. Conf. Architectural Support Program. Lang. Operating Syst. (ASPLOS)*, 2018, pp. 461–475.

[21] Y. Li, A. Louri, and A. Karanth, "Scaling deep-learning inference with chiplet-based architecture and photonic interconnects," in *Proc. 58th ACM/IEEE Design Autom. Conf. (DAC)*, Dec. 2021, pp. 931–936.

[22] Y. Li, A. Louri, and A. Karanth, "SPRINT: A high-performance, energy-efficient, and scalable chiplet-based accelerator with photonic interconnects for CNN inference," *IEEE Trans. Parallel Distrib. Syst.*, vol. 33, no. 10, pp. 2332–2345, Oct. 2022.

[23] Y. Li, A. Louri, and A. Karanth, "SPACX: Silicon photonics-based scalable chiplet accelerator for DNN inference," in *Proc. IEEE Int. Symp. High-Performance Comput. Archit. (HPCA)*, 2022, pp. 831–845.

[24] Y. Demir, Y. Pan, S. Song, N. Hardavellas, J. Kim, and G. Memik, "Galaxy: A high-performance energy-efficient multi-chip architecture using photonic interconnects," in *Proc. 28th ACM Int. Conf. Supercomput. (ICS)*, 2014, pp. 303–312.

[25] P. Grani, R. Proietti, V. Akella, and S. J. B. Yoo, "Design and evaluation of AWGR-based photonic NoC architectures for 2.5D integrated high performance computing systems," in *Proc. IEEE Int. Symp. High Perform. Comput. Archit. (HPCA)*, Feb. 2017, pp. 289–300.

[26] Y.-H. Kao and H. J. Chao, "BLOCON: A bufferless photonic clos network-on-chip architecture," in *Proc. 5th ACM/IEEE Int. Symp. Netw.-on-Chip (NOCS)*, May 2011, pp. 81–88.

[27] N. Kirman *et al.*, "Leveraging optical technology in future bus-based chip multiprocessors," in *Proc. 39th Annu. IEEE/ACM Int. Symp. Microarchitecture (MICRO)*, Dec. 2006, pp. 492–503.

[28] C. Li, M. Browning, P. V. Gratz, and S. Palermo, "LumiNOC: A power-efficient, high-performance, photonic network-on-chip," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 33, no. 6, pp. 826–838, Jun. 2014.

[29] A. Narayan, Y. Thonnart, P. Vivet, and A. K. Coskun, "PROWAVES: Proactive runtime wavelength selection for energy-efficient photonic NoCs," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 40, no. 10, pp. 2156–2169, Oct. 2021.

[30] A. Narayan, Y. Thonnart, P. Vivet, A. Joshi, and A. K. Coskun, "System-level evaluation of chip-scale silicon photonic networks for emerging data-intensive applications," in *Proc. Design, Autom. Test Eur. Conf. Exhib. (DATE)*, Mar. 2020, pp. 1444–1449.

[31] A. Narayan, Y. Thonnart, P. Vivet, C. F. Tortolero, and A. K. Coskun, "WAVES: Wavelength selection for power-efficient 2.5D-integrated photonic NoCs," in *Proc. Design, Autom. Test Eur. Conf. Exhib. (DATE)*, Mar. 2019, pp. 516–521.

[32] Y. Pan, P. Kumar, J. Kim, G. Memik, Y. Zhang, and A. Choudhary, "Firefly: Illuminating future network-on-chip with nanophotonics," in *Proc. 36th Annu. Int. Symp. Comput. Archit. (ISCA)*, 2009, pp. 429–440.

[33] Y. Thonnart *et al.*, "POPSTAR: A robust modular optical NoC architecture for chiplet-based 3D integrated systems," in *Proc. Design, Autom. Test Eur. Conf. Exhib. (DATE)*, Mar. 2020, pp. 1456–1461.

[34] D. Vantrease *et al.*, "Corona: System implications of emerging nanophotonic technology," in *Proc. Int. Symp. Comput. Archit.*, Jun. 2008, pp. 153–164.

[35] A. K. K. Ziabari, J. L. Abellán, R. Ubal, C. Chen, A. Joshi, and D. Kaeli, "Leveraging silicon-photonic NoC for designing scalable GPUs," in *Proc. 29th ACM Int. Conf. Supercomput.*, Jun. 2015, pp. 273–282.

[36] L. Cavigelli, D. Gschwend, C. Mayer, S. Willi, B. Muheim, and L. Benini, "Origami: A convolutional network accelerator," in *Proc. 25th Ed., Great Lakes Symp. VLSI*, May 2015, pp. 199–204.

[37] S. Chakradhar, M. Sankaradas, V. Jakkula, and S. Cadambi, "A dynamically configurable coprocessor for convolutional neural networks," in *Proc. 37th Annu. Int. Symp. Comput. Archit. (ISCA)*, 2010, pp. 247–257.

[38] T. Chen *et al.*, "DianNao: A small-footprint high-throughput accelerator for ubiquitous machine-learning," in *Proc. ACM Int. Conf. Architectural Support Program. Lang. Operating Syst. (ASPLOS)*, pp. 269–284, 2014.

[39] Y.-H. Chen, J. Emer, and V. Sze, "Eyeriss: A spatial architecture for energy-efficient dataflow for convolutional neural networks," in *Proc. ACM/IEEE 43rd Annu. Int. Symp. Comput. Archit. (ISCA)*, Jun. 2016, pp. 367–379.

[40] Z. Du *et al.*, "ShiDianNao: Shifting vision processing closer to the sensor," in *Proc. 42nd Annu. Int. Symp. Comput. Archit.*, Jun. 2015, pp. 92–104.

[41] S. Park, K. Bong, D. Shin, J. Lee, S. Choi, and H.-J. Yoo, "4.6 A1.93TOPS/W scalable deep learning/inference processor with tetra-parallel MIMD architecture for big-data applications," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, Feb. 2015, pp. 80–81.

[42] M. Sankaradas *et al.*, "A massively parallel coprocessor for convolutional neural networks," in *Proc. 20th IEEE Int. Conf. Appl.-Specific Syst., Architectures Processors*, Jul. 2009, pp. 53–60.

[43] M. Gao, J. Pu, X. Yang, M. Horowitz, and C. Kozyrakis, "TETRIS: Scalable and efficient neural network acceleration with 3D memory," in *Proc. ACM Int. Conf. Architectural Support Program. Lang. Operating Syst. (ASPLOS)*, 2017, pp. 751–764.

[44] X. Yang *et al.*, "Interstellar: Using halide's scheduling language to analyze DNN accelerators," in *Proc. 25th Int. Conf. Architectural Support Program. Lang. Operating Syst.*, Mar. 2020, pp. 369–383.

[45] R. Marchetti, C. Lacava, L. Carroll, K. Gradkowski, and P. Minzioni, "Coupling strategies for silicon photonics integrated chips," *Photon. Res.*, vol. 7, no. 2, pp. 201–239, 2019.

[46] W. Bogaerts *et al.*, "Silicon microring resonators," *Laser Photon. Rev.*, vol. 6, no. 1, pp. 47–73, Jan. 2012.

[47] R. Morris, A. K. Kodi, and A. Louri, "Dynamic reconfiguration of 3D photonic networks-on-chip for maximizing performance and improving fault tolerance," in *Proc. 45th Annu. IEEE/ACM Int. Symp. Microarchitecture*, Dec. 2012, pp. 282–293.

[48] S. Van Winkle, A. K. Kodi, R. Bunescu, and A. Louri, "Extending the power-efficiency and performance of photonic interconnects for heterogeneous multicores with machine learning," in *Proc. IEEE Int. Symp. High Perform. Comput. Archit. (HPCA)*, Feb. 2018, pp. 480–491.

[49] S. Werner, J. Navaridas, and M. Lujan, "Designing low-power, low-latency networks-on-chip by optimally combining electrical and optical links," in *Proc. IEEE Int. Symp. High Perform. Comput. Archit. (HPCA)*, Feb. 2017, pp. 265–276.

[50] E. Peter, A. Thomas, A. Dhawan, and S. R. Sarangi, "Active microring based tunable optical power splitters," *Opt. Commun.*, vol. 359, pp. 311–315, Jan. 2016.

[51] J. Bashir, E. Peter, and S. R. Sarangi, "A survey of on-chip optical interconnects," *ACM Comput. Surv.*, vol. 51, no. 6, pp. 1–34, 2019.

[52] H. Kwon, P. Chatarasi, V. Sarkar, T. Krishna, M. Pellauer, and A. Parashar, "MAESTRO: A data-centric approach to understand reuse, performance, and hardware cost of DNN mappings," *IEEE Micro*, vol. 40, no. 3, pp. 20–29, May 2020.

[53] P. Rosenfeld, E. Cooper-Balis, and B. Jacob, "DRAMSim2: A cycle accurate memory system simulator," *Comput. Archit. Lett.*, vol. 10, no. 1, pp. 16–19, Jan./Jun. 2011.

[54] H. Jayatilleka, M. Caverley, N. A. F. Jaeger, S. Shekhar, and L. Chrostowski, "Crosstalk limitations of microring-resonator based WDM demultiplexers on SOI," in *Proc. IEEE Opt. Interconnects Conf. (OI)*, Apr. 2015, pp. 48–49.

[55] C. Sun *et al.*, "DSENT—A tool connecting emerging photonics with electronics for opto-electronic networks-on-chip modeling," in *Proc. IEEE/ACM 6th Int. Symp. Networks-on-Chip*, May 2012, pp. 201–210.

[56] R. W. Morris and A. K. Kodi, "Power-efficient and high-performance multi-level hybrid nanophotonic interconnect for multicores," in *Proc. 4th ACM/IEEE Int. Symp. Netw.-on-Chip*, 2010, pp. 207–214.

[57] G. Li *et al.*, "25 Gb/s 1 V-driving CMOS ring modulator with integrated thermal tuning," *Opt. Exp.*, vol. 19, no. 21, pp. 20435–20443, 2011.

[58] S. Pasricha and S. Bahirat, "OPAL: A multi-layer hybrid photonic NoC for 3D ICs," in *Proc. 16th Asia South Pacific Design Autom. Conf. (ASP-DAC)*, Jan. 2011, pp. 345–350.
[59] M. Behadori, M. Nikdast, Q. Cheng, and K. Bergman, "Universal design of waveguide bends in silicon-on-insulator photonics platform," *J. Lightw. Technol.*, vol. 37, no. 10, pp. 3044–3054, Jul. 1, 2019.
[60] A. Biberman *et al.*, "Photonic network-on-chip architectures using multilayer deposited silicon materials for high-performance chip multiprocessors," *ACM J. Emerg. Technol. Comput. Syst.*, vol. 7, no. 2, pp. 1–25, 2011.
[61] A. Joshi *et al.*, "Silicon-photonic clos networks for global on-chip communication," in *Proc. 3rd ACM/IEEE Int. Symp. Netw.-on-Chip*, May 2009, pp. 124–133.
[62] N. Muralimanohar, R. Balasubramonian, and N. P. Jouppi, "CACTI 6.0: A tool to model large caches," *HP Laboratories*, vol. 27, pp. 1–24, Apr. 2009.
[63] R. Polster, Y. Thonnart, G. Waltener, J.-L. Gonzalez, and E. Cassan, "Efficiency optimization of silicon photonic links in 65-nm CMOS and 28-nm FDSOI technology nodes," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 24, no. 12, pp. 3450–3459, Dec. 2016.
[64] C. DeCusatis, *Handbook of Fiber Optic Data Communication: A Practical Guide to Optical Networking*. New York, NY, USA: Academic, 2013.
[65] A. V. Krishnamoorthy *et al.*, "Computer systems based on silicon photonic interconnects," *Proc. IEEE*, vol. 97, no. 7, pp. 1337–1361, Jul. 2009.
[66] Y. Thonnart *et al.*, "A 10 Gb/s Si-photonic transceiver with 150 μW 120 μs-lock-time digitally supervised analog microring wavelength stabilization for 1 Tb/s/mm^2 die-to-die optical networks," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, Feb. 2018, pp. 350–352.
[67] G. Li *et al.*, "40 Gb/s thermally tunable CMOS ring modulator," in *Proc. 9th Int. Conf. Group IV Photon. (GFP)*, Aug. 2012, pp. 1–3.
[68] H. Zheng, K. Wang, and A. Louri, "A versatile and flexible chiplet-based system design for heterogeneous manycore architectures," in *Proc. 57th ACM/IEEE Design Autom. Conf. (DAC)*, Jul. 2020, pp. 1–6.

Yuan Li (Student Member, IEEE) received the B.S. degree in physics from the University of Science and Technology of China in 2010, and the M.S. degree in microelectronics from the University of Newcastle upon Tyne in 2011. He is currently pursuing the Ph.D. degree in computer engineering with George Washington University. His research interests include machine learning architectures, accelerator-rich heterogeneous systems, and emerging interconnect and memory technologies.

Ke Wang (Student Member, IEEE) received the B.S. degree in electrical engineering from Peking University in 2013, and the M.S. degree in electrical engineering from the Worcester Polytechnic Institute in 2015. He is currently pursuing the Ph.D. degree in computer engineering with the School of Engineering and Applied Science, George Washington University. His research interests include high-performance, energy-efficient, fault-tolerant, and secure interconnection networks.

Hao Zheng (Student Member, IEEE) received the B.S. degree in electrical engineering from Beijing Jiaotong University, Beijing, China, and the M.S. degree in electrical engineering from George Washington University, Washington, DC, USA, where he is currently pursuing the Ph.D. degree in computer engineering. His research interests include computer architecture and parallel computing, with emphasis on interconnection networks, machine learning techniques for efficient computing, and energy-efficient manycore architecture designs.

Ahmed Louri (Fellow, IEEE) received the Ph.D. degree in computer engineering from the University of Southern California, Los Angeles, CA, USA, in 1988. From 1988 to 2015, he was a Professor of electrical and computer engineering at The University of Arizona, and during that time, he worked six years (from 2000 to 2006) as the Chair of the Computer Engineering Program. From 2010 to 2013, he served as a Program Director for the National Science Foundation's (NSF) Directorate for Computer and Information Science and Engineering. He is currently the David and Marilyn Karlgaard Endowed Chair Professor of electrical and computer engineering at George Washington University, where he joined in August 2015. He is also the Director of the High Performance Computing Architectures and Technologies Laboratory. He directed the core computer architecture program and was on the management team of several cross-cutting programs. He conducts research in the broad area of computer architecture and parallel computing, with emphasis on interconnection networks, optical interconnects for scalable parallel computing systems, reconfigurable computing systems, and power-efficient and reliable network-on-chips (NoCs) for multicore architectures. Recently, he has been concentrating on: energy-efficient, reliable, and high-performance many-core architectures; accelerator-rich reconfigurable heterogeneous architectures; machine learning techniques for efficient computing, memory, and interconnect systems; emerging interconnect technologies (photonic, wireless, RF, and hybrid) for NoCs; future parallel computing models and architectures (including convolutional neural networks, deep neural networks, and approximate computing); and cloud computing and data centers. He was a recipient of the 2020 IEEE Computer Society Edward J. McCluskey Technical Achievement Award, "for pioneering contributions to the solution of on-chip and off-chip communication problems for parallel computing and manycore architectures." He is also the Editor-in-Chief of the IEEE TRANSACTIONS ON COMPUTERS. More information can be found at: https://hpcat.seas.gwu.edu/Director.html.

Avinash Karanth (Senior Member, IEEE) received the B.E. degree in electronics and communications from the Manipal Institute of Technology, Mangalore University, in February 2000, and the M.S. and Ph.D. degrees from the Electrical and Computer Engineering Department, The University of Arizona, in May 2003 and August 2006, respectively. He is currently the Joseph Jachinowski Professor with the School of Electrical Engineering and Computer Science, Ohio University, Athens, OH, USA. He directs the Technologies for Emerging Computer Architecture Laboratory (TEAL), Ohio University. His research interests include computer architecture, optical interconnects, network-on-chips (NoCs), and emerging technologies, such as nanophotonics, 3D, and wireless interconnects. He is a member of ACM. He was a recipient of the NSF CAREER Award in 2011, the Presidential Research Scholar Award in 2017, the Best Paper Award at the ICCD 2013 Conference, and his papers have been nominated for Best Paper at the IEEE Symposium on NoCs in May 2010 and the IEEE Asia & South Pacific Design Automation Conference (ASP-DAC) in January 2009.