Scaling Deep-Learning Inference with Chiplet-based Architecture and Photonic Interconnects

Yuan Li*, Ahmed Louri*, Avinash Karanth†
*Department of Electrical and Computer Engineering, George Washington University, Washington, DC, USA
†School of Electrical Engineering and Computer Science, Ohio University, Athens, Ohio, USA
Emails: *{liyuan5859, louri}@gwu.edu, †karanth@ohio.edu

***Abstract*—Chiplet-based architectures have been proposed to scale computing systems for deep neural networks (DNNs). Prior work has shown that for the chiplet-based DNN accelerators, the electrical network connecting the chiplets poses a major challenge to system performance, energy consumption, and scalability. Some emerging interconnect technologies such as silicon photonics can potentially overcome the challenges facing electrical interconnects as photonic interconnects provide high bandwidth density, superior energy efficiency, and ease of implementing broadcast and multicast operations that are prevalent in DNN inference. In this paper, we propose a chiplet-based architecture named SPRINT for DNN inference. SPRINT uses a global buffer to simplify the data transmission between storage and computation, and includes two novel designs: (1) a reconfigurable photonic network that can support diverse communications in DNN inference with minimal implementation cost, and (2) a customized dataflow that exploits the ease of broadcast and multicast feature of photonic interconnects to support highly parallel DNN computations. Simulation studies using ResNet-50 DNN model show that SPRINT achieves 46% and 61% execution time and energy consumption reduction, respectively, as compared to other state-of-the-art chiplet-based architectures with electrical or photonic interconnects.**

***Keywords*—chiplet, accelerator, silicon photonics, deep learning**

I. Introduction

Modern deep neural networks (DNNs) achieve high inference accuracy by increasing their model size [1]–[4]. As the model size exceeds the computation and storage capacity of one single chip, the chiplet-based architectures [5]–[7] have been shown to be a viable approach to scale up the system. However, for such chiplet-based DNN accelerators, the electrical inter-chiplet network poses a major challenge to system performance, energy consumption, and scalability.

The emerging silicon photonics technology has the potential to improve bandwidth density, energy efficiency, and overall performance of chiplet-based architectures [8]–[12]. The distance-independent property of photonics allows similar energy consumption for data transmission to different chiplets, albeit with marginal timing variations. This simplifies energy and timing constraints for DNN inference since data can be transmitted simultaneously without complex orchestration as required in electrical networks. Further, photonic interconnects have the capability of adequately supporting a multitude of communications such as one-to-many (multicast) and one-to-all (broadcast) which are prevalent in DNN inference. While there have been a few chiplet-based architectures with photonic interconnects [8], [11], they often provide full connectivity and uniform bandwidth between chiplets via crossbar interconnects for all-to-all communication. Such crossbar-based networks are costly and difficult to scale for larger systems.

In this paper, we propose **SPRINT** - a chiplet-based architecture with **S**ilicon **P**hotonic **R**econfigurable **INT**erconnects for DNN inference applications. SPRINT includes two novel designs: (1) a reconfigurable photonic inter-chiplet network that connects the global buffer (GB) and other accelerator chiplets and supports diverse communications in DNN inference, and (2) a customized dataflow that exploits the ease of broadcast and multicast feature of photonic interconnects to support highly parallel DNN computations. Specifically, the GB simplifies the all-to-all communication [5] into three categories: unicast communication from the GB to other accelerator chiplets for weight transmission, multicast or broadcast communication from the GB to other accelerator chiplets for input activation (IA) transmission, and unicast communication from each accelerator chiplet to the GB for partial sum transmission. The proposed photonic inter-chiplet network is carefully designed to support all three communication categories with minimal overhead through network reconfiguration. We compare SPRINT with other state-of-the-art chiplet-based architectures [5], [8] with either electrical or photonic inter-chiplet interconnects using the ResNet-50 DNN model [1]. Simulation studies show that SPRINT achieves 46% and 61% reduction in execution time and energy consumption, respectively. SPRINT also exhibits better scalability as the number of accelerator chiplets increases.

II. SPRINT Architecture

A. Architecture Overview

Fig. 1 presents a small-scale SPRINT architecture with a GB, four accelerator chiplets, and a photonic inter-chiplet network implemented on a photonic interposer.

GB: The GB is implemented on a separate silicon die. It includes four modules: GB memory banks for temporary data storage, a computing engine to perform non-convolution operations (e.g., bias, activation, pooling, etc.), peripheral circuitries of transmitters and receivers that interact with corresponding photonic components on the photonic interposer, and a near-data accumulation engine (NAE). NAE takes the partial sums received from accelerator chiplets as inputs and performs partial sum accumulation operations. The outputs

978-1-6654-3274-0/21/$31.00 ©2021 IEEE

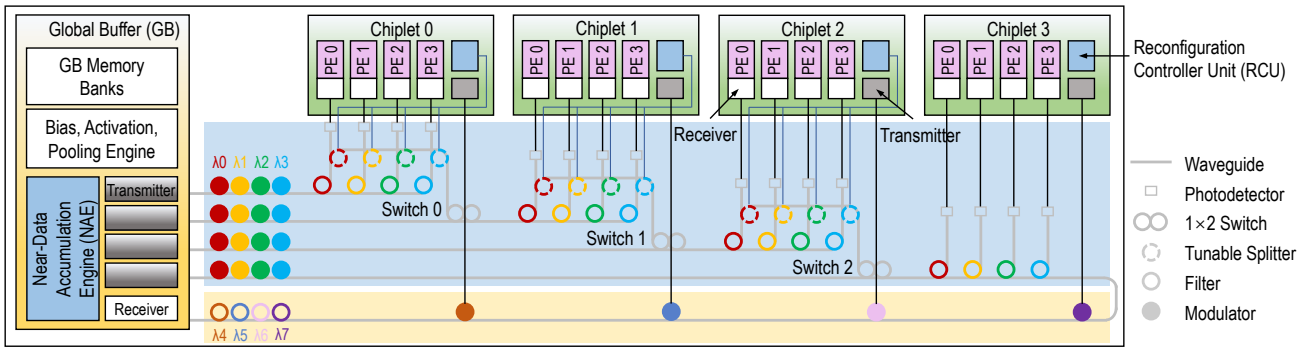

Fig. 1: SPRINT architecture overview. We assume four chiplets and four PEs per chiplet. Eight wavelengths are multiplexed in the photonic network. $\lambda 0$, $\lambda 1$, $\lambda 2$, $\lambda 3$ are responsible for data transmission from the GB to all accelerator chiplets, while $\lambda 4$, $\lambda 5$, $\lambda 6$, $\lambda 7$ are responsible for data transmission from a corresponding accelerator chiplet to the GB.

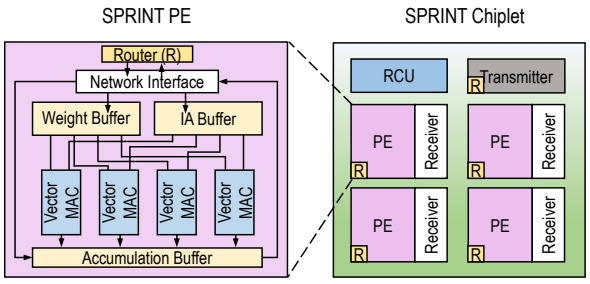

Fig. 2: PE and chiplet of the SPRINT architecture. Within a chiplet, an electrical network-on-chip (2D mesh in our case) is used for data transmission between PEs and the transmitter.

of NAE are either forwarded to the computing engine for non-convolution operations or stored in the GB memory banks.

Accelerator Chiplet: As shown in Fig. 2, each accelerator chiplet consists of a PE array, a reconfigurable control unit (RCU), a transmitter, and several receivers that are distributed to processing elements (PEs). Within each PE, there are several vector multiply-accumulate (MAC) units and dedicated buffers for temporary storage of weights, IAs, and partial sums. RCU is responsible for tuning specific photonic components (we discuss the details in the following section) during the network reconfiguration process. The transmitter and receivers interact with the corresponding photonic components on the photonic interposer for data transmission.

Photonic Inter-chiplet Network: The photonic inter-chiplet network is implemented on the photonic interposer. As shown in Fig. 1, the upper half of this network can be configured to dedicated GB-to-chiplet channels, a single-write-multiple-read (SWMR) channel, or several segmented SWMR channels for data transmission from the GB to accelerator chiplets. The lower half of this network works as a multiple-write-single-read (MWSR) channel for data transmission from accelerator chiplets to the GB. Accordingly, the wavelengths multiplexed in the network are divided into two groups - one group ($\lambda 0$, $\lambda 1$, $\lambda 2$ and $\lambda 3$ in Fig. 1) is utilized in the upper half of the network while the other group ($\lambda 4$, $\lambda 5$, $\lambda 6$ and $\lambda 7$ in Fig. 1) is utilized in the lower half of the network. The color of a micro-ring resonator (MRR) shown in Fig. 1 indicates the wavelength on which this MRR works.

B. SPRINT Intra-chiplet Network

As SPRINT architecture focuses on package-level integration and communication bottleneck alleviation, we assume a similar chiplet architecture as in [5] and a 2D electrical mesh as the intra-chiplet network. We adopt wormhole flow control and XY routing. This intra-chiplet network is responsible for data transmission between local PEs and also from the local PEs to the transmitter, which then drives the corresponding photonic modulator on the interposer to transmit data to the GB.

C. SPRINT Inter-chiplet Network

The photonic inter-chiplet network is utilized to connect the GB and PEs on accelerator chiplets. In the example shown in Fig. 1, wavelengths $\lambda 0$, $\lambda 1$, $\lambda 2$ and $\lambda 3$ are responsible for data transmission from the GB to accelerator chiplets while $\lambda 4$, $\lambda 5$, $\lambda 6$ and $\lambda 7$ are responsible for data transmission from accelerator chiplets to the GB. By directly transmitting weights and IAs from the GB to accelerator chiplets and performing partial sum accumulation operations at NAE, data transmission between any accelerator chiplets is eliminated.

Key Components: There are two key components in the proposed photonic network: tunable splitter [13] and electrical-optical switch [14]. A tunable splitter achieves different split ratios on a specific wavelength when different bias voltages are applied to its MRR. The MRR works in the transient zone between on-resonance and off-resonance so that a fraction of the laser power is guided to the corresponding photodetector while the rest is forwarded to the downstream accelerator chiplets, as shown in Fig. 1. The electrical-optical switch (1×2 switch in Fig. 1) associated with a waveguide determines whether it works as a dedicated GB-to-chiplet channel or is combined with other waveguides to form an SWMR channel.

GB to Chiplet Communication: The upper half of the photonic network shown in Fig. 1 is responsible for data transmission from the GB to accelerator chiplets, and can be configured into one of the three modes: unicast mode, broadcast mode, and multicast mode. In the unicast mode, all the electrical-optical switches are at off-resonance and all the tunable splitters are disabled. Each of the four waveguides between the GB and accelerator chiplets works independently. For example, the GB can transmit data to *Chiplet* 0 by

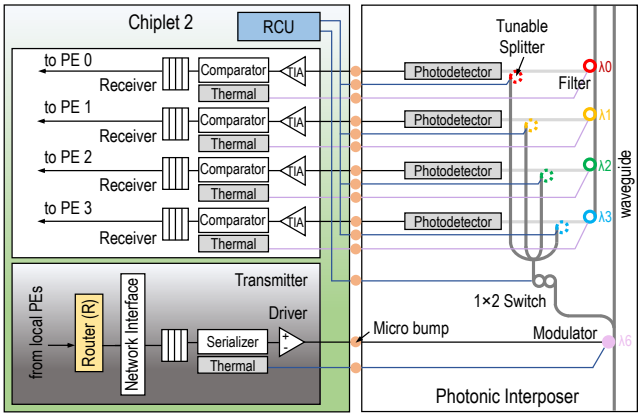

Fig. 3: Physical implementation of the photonic inter-chiplet network, taking the part of $Chiplet$ 2 in Fig. 1 as an example.

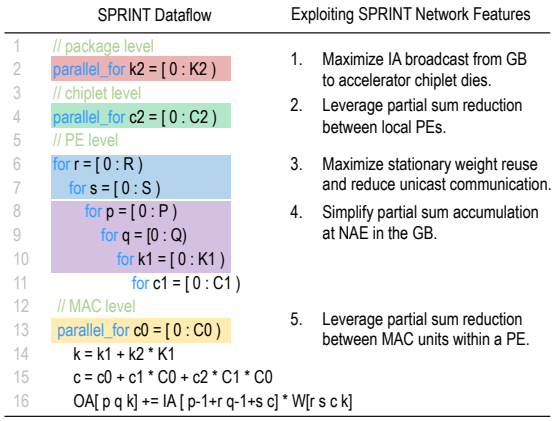

Fig. 4: SPRINT customized dataflow that exploits the photonic inter-chiplet network.

modulating wavelengths $\lambda 0$, $\lambda 1$, $\lambda 2$, and $\lambda 3$ on the first waveguide. Meanwhile, the GB can also transmit data to $Chiplet$ 1 by modulating the same four wavelengths on the second waveguide, as the electrical-optical switch between these two waveguides is at off-resonance and the two waveguides work as independent channels. The unicast mode is used to distribute weights from the GB to the weight buffer in each PE.

In the broadcast mode, all electrical-optical switches are at on-resonance and all the tunable splitters are tuned to appropriate split ratios based on their positions in the network. Thus, all four waveguides in Fig. 1 are combined into an SWMR broadcast channel. By using only one set of modulators, data can be broadcast from the GB to all accelerator chiplets. Please note that the correlation between a wavelength and a PE still exists (e.g., $\lambda 0$ is used to broadcast from the GB to PEs labeled PE 0 in all accelerator chiplets). The broadcast mode is used to broadcast IAs from the GB to the IA buffer of each PE.

The multicast mode is a combination of the above two modes. In the multicast mode, the waveguides in the upper half of the photonic network are grouped to form several segmented SWMR channels. For example, as shown in Fig. 1, by tuning $Switch$ 0 and $Switch$ 2 at on-resonance and $Switch$ 1 at off-resonance, we create two segmented SWMR channels: one for multicast from the GB to $Chiplet$ 0 and $Chiplet$ 1 while the other one for multicast from the GB to $Chiplet$ 2 and $Chiplet$ 3. The multicast mode provides the flexibility to support more sophisticated dataflows or simultaneous execution of multiple DNN layers.

Chiplet to GB Communication: The lower half of the photonic network shown in Fig. 1 is responsible for data transmission from accelerator chiplets to the GB, and works as an MWSR channel to collect partial sums. Each accelerator chiplet is assigned a specific wavelength (e.g., $\lambda 4$ is assigned to $Chiplet$ 0) for data transmission to the GB. Please note that it is possible to assign several wavelengths to an accelerator chiplet if the bandwidth demand of an accelerator chiplet exceeds the data rate of a single wavelength.

Physical Implementation: Light generated by the off-chip laser source is coupled to the waveguides on the photonic interposer through couplers. The peripheral circuitries of the transmitter and receivers on each chiplet are connected to corresponding photonic components on the photonic interposer through micro-bumps. Fig. 3 shows the physical implementation of one accelerator chiplet ($Chiplet$ 2 shown in Fig. 1) and the related photonic components. The vertical and horizontal waveguides in Fig. 3 are in separate layers to avoid crossing.

D. SPRINT Dataflow

Fig. 4 describes the customized SPRINT dataflow and how it leverages the proposed photonic inter-chiplet network. This dataflow is derived from the weight-stationary dataflow [5]. If we assume a different accelerator chiplet architecture with a different dataflow (e.g., row-stationary dataflow [3]), SPRINT dataflow may change accordingly. However, the primary target of SPRINT dataflow remains as maximizing the broadcast and multicast communication.

Package-level Data Partition: At the package level, weights are logically partitioned along the output channel k dimension and physically mapped to different accelerator chiplets ($K2 \leqslant$ number of accelerator chiplets). As weight planes from different k dimensions may take the same IA plane as input to perform convolution operations, this IA plane is broadcast to accelerator chiplets allocated with weights of different k dimension values using the broadcast mode. For example, PE 0 of $Chiplet$ 0 and PE 0 of $Chiplet$ 1 in Fig. 1 perform convolution operations on the same IA plane which is broadcast to these two PEs using $\lambda 0$. By doing so, we minimize the number of involved modulators and leverage the ease of broadcast feature of the photonic inter-chiplet network over other electrical counterparts. If the number of output channels K is smaller than the number of accelerator chiplets, we may process the convolution layer with a subset of accelerator chiplets or process multiple convolution layers in parallel. In both cases, we use the multicast mode to minimize the number of modulators involved.

Chiplet-level Data Partition: At the chiplet level, weights are logically partitioned along the input channel c dimension and physically mapped to different PEs ($C2 \leqslant$ number of PEs per chiplet) in each accelerator chiplet. As weight planes

TABLE I: SPRINT Architecture Parameters

Package	Number of chiplets	64
	Global buffer size per chiplet	64 KiB
	Data rate per wavelength	10 Gbps
	Inter-Chiplet Network Bandwidth	100 GB/s/chiplet
Chiplet	Number of PEs	64
	Intra-Chiplet network	2D mesh
	Intra-Chiplet network bandwidth	68 GB/s/PE
PE	Weight buffer size	32 KiB
	IA buffer size	8 KiB
	Accumulation buffer size	3 KiB
	Vector MAC width	8
	Number of vector MACs	8

from different c dimensions take different IA planes as input to perform convolution operations, different IA planes are transmitted to PEs allocated with weights of different c dimension values using different wavelengths. As shown in Fig. 1, PE 0 and PE 1 of $Chiplet$ 0 perform convolution operations on two separate IA planes which are transmitted to the two PEs using $\lambda 0$ and $\lambda 1$, respectively. Data partitions at the package and chiplet levels determine the pattern of weight and IA transmission from the GB to accelerator chiplets. As SPRINT dataflow is developed from the weight-stationary dataflow [5], weights are allocated to PEs without overlap and transmitted using the unicast mode.

PE-level Dataflow: There are six loops in the PE-level dataflow. The outer two loops at the r (height of weight filter) and s (width of weight filter) dimensions maximize the reuse of the stationary weights. The middle three loops at the p (height of output activation plane), q (width of output activation plane), and k dimensions simplify the partial sum accumulation process at NAE because the partial sums have been fully accumulated temporally ($c1$ loop) and spatially ($c0$ and $c2$ loops) before being transmitted to NAE.

MAC-level Dataflow: Within each PE, we perform dot productions along the c dimension to exploit efficient accumulation of locally generated partial sums as in [5].

III. Evaluation Methodology

In this section, we discuss the methodologies to compared SPRINT with other state-of-the-art baselines in terms of performance and energy consumption. Table I lists some key architectural parameters of SPRINT. For SPRINT and other baselines, we assume the same chiplet architecture as shown in Table I except that the GB is implemented in a single silicon die in SPRINT while equally distributed to accelerator chiplets in other baselines. Please note that we choose 28 nm technology to implement the photonic inter-chiplet network [15] as it is common to adopt a relatively less-advanced technology for the fabrication of the photonic interposer.

Simulator: We utilize a customized version of the open-source Timeloop simulator [16]. We extend this simulator to support the non-uniform distribution of latency and bandwidth between PEs. In order to obtain execution time results, the simulator monitors the number of arithmetic operations and the number of accesses to each memory hierarchy, taking the dataflow and system configuration parameters into account. The number of arithmetic operations is used to calculate the computation time, while the number of accesses to each on-package memory hierarchy is used to calculate the on-package communication time. We take the hierarchical network topology and bandwidth limit into account when calculating the on-package communication time. The off-package communication time is obtained from the DRAMSim2 simulator [17]. The overall execution time is derived by adding up the computation time, the on-package communication time, and the off-package communication time, considering the overlap caused by the buffering of the GB and other memory hierarchies.

Power Model: The power consumption of arithmetic operations is modeled by *Synopsys Design Compiler*. The power consumption numbers of accessing on-package memory hierarchies (local buffers in each PE and the GB) and off-package DRAM are obtained from CACTI 6.0 [22] and DRAMSim2 simulators, respectively. The power consumption of data transmission through an electrical interconnect is obtained from DSENT [21]. Finally, the power consumption of data transmission through a photonic link is derived from (1):

$$P_{total} = P_{laser} + P_{TX} + P_{RX} \quad (1)$$

P_{total} includes three parts: laser power P_{laser}, transmitter circuitry power P_{TX}, and receiver circuitry power P_{RX}. We calculate P_{TX} and P_{RX} using parameters in [15] and scale the results to 28 nm technology [15], [23]. P_{laser} can be further divided into three parts: photodetector sensitivity P_{rs}, insertion loss C_{loss}, and system margin M_{system} as shown in (2):

$$P_{laser} = P_{rs} + C_{loss} + M_{system} \quad (2)$$

We assume a 4 dB [24] system margin to account for the additional power loss. P_{rs} and C_{loss} are obtained or derived from parameters listed in Table II. We assume four 90-degree bends for each waveguide. The length of a waveguide is set at 10 cm. We obtain the energy consumption of 0.77 pJ/bit in SPRINT photonic inter-chiplet network.

Baselines: We compare SPRINT with two prior designs, Simba [5] and POPSTAR [8]. For fair comparison, Simba and POPSTAR are scaled to include the same type and number of accelerator chiplets as in SPRINT. Simba is specifically designed for DNN inference acceleration with an electrical inter-chiplet network (we assume 100 $GB/s/chiplet$ bandwidth and per-hop latency of 10 clock cycles). POPSTAR is developed for more general applications but with a 2.5D-integrated photonic inter-chiplet network (we assume the same latency and per-chiplet bandwidth as in SPRINT). SPRINT architecture is a chiplet-based DNN accelerator (similar to Simba) with a photonic inter-chiplet network (similar to POPSTAR).

Benchmark: We employ the ResNet-50 [1] DNN model as the evaluation benchmark. Table III lists the 21 different layer configurations obtained from ResNet-50. We include the notation, the layer name in *Caffe*, and the layer configurations (H: IA plane size, C: number of input channels, R: weight filter size, K: number of output channels, S: stride) in this table. Please note that some layers in ResNet-50 share the same configuration (e.g., res2a_branch1 layer and res2[a-c]_branch2c

TABLE II: Photonic Parameters

Component	Value
Laser source	5 dB [18]
Coupler	1 dB [18]
Waveguide	1 dB/cm [18]
Splitter	0.2 dB [19]
Bend	1 dB [20]
Waveguide crossover	0.05 dB [20]
Modulator loss	1 dB [21]
Ring through	0.01 dB [21]
Photodetector	0.1 dB [18]
Waveguide-to-receiver	0.5 dB [20]
Receiver sensitivity	-26 dBm [18]

TABLE III: Convolutional Layers with Different Parameters in ResNet-50

Notation	Layer(s)	H	C	R	K	S	Notation	Layer(s)	H	C	R	K	S
L1	conv1	224,	3,	7,	64,	2	L12	res4a_branch2a	28,	512,	1,	256,	2
L2	res2a_branch2a	56,	64,	1,	64,	1	L13	res4[a-f]_branch2b	14,	256,	3,	256,	1
L3	res2[a-c]_branch2b	56,	64,	3,	64,	1	L14	res4[a-f]_branch2c	14,	256,	1,	1024,	1
L4	res2[a-c]_branch2c	56,	64,	1,	256,	1	L15	res4[b-f]_branch2a	14,	1024,	1,	256,	1
L5	res2[b-c]_branch2a	56,	256,	1,	64,	1	L16	res5a_branch1	14,	1024,	1,	2048,	2
L6	res3a_branch1	56,	256,	1,	512,	2	L17	res5a_branch2a	14,	1024,	1,	512,	2
L7	res3a_branch2a	56,	256,	1,	128,	2	L18	res5[a-c]_branch2b	7,	512,	3,	512,	1
L8	res3[a-d]_branch2b	28,	128,	3,	128,	1	L19	res5[a-c]_branch2c	7,	512,	1,	2048,	1
L9	res3[a-d]_branch2c	28,	128,	1,	512,	1	L20	res5[b-c]_branch2a	7,	2048,	1,	512,	1
L10	res3[b-d]_branch2a	28,	512,	1,	128,	1	L21	fc1000	1,	2048,	1,	1000,	1
L11	res4a_branch1	28,	512,	1,	1024,	2							

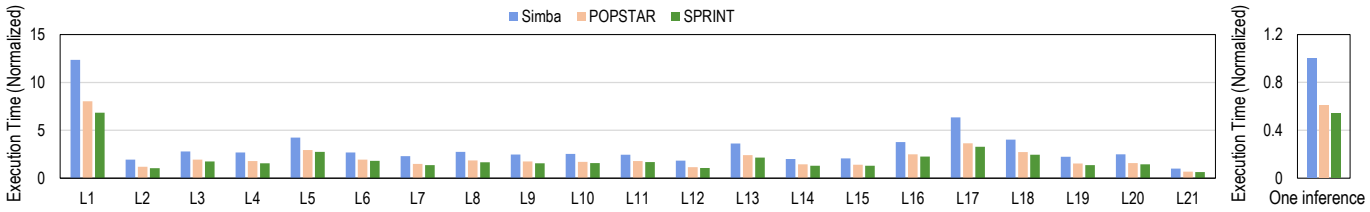

Fig. 5: Per-layer execution time (normalized to L21 in Simba) and execution time of one inference pass (normalized to Simba).

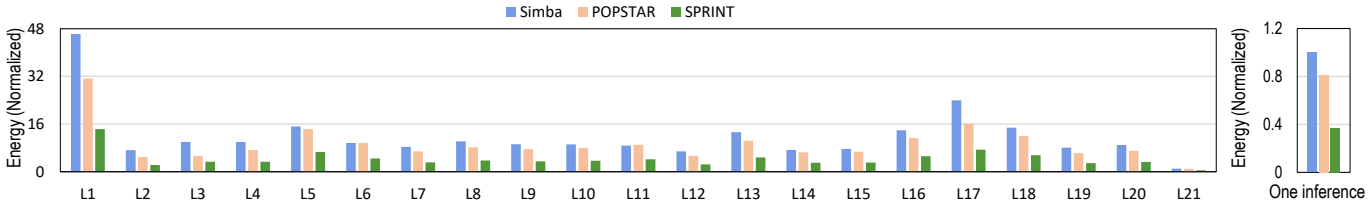

Fig. 6: Per-layer energy consumption (normalized to L21 in Simba) and energy of one inference pass (normalized to Simba).

layer). We compare SPRINT and two other baselines in a layer-by-layer manner and then accumulate per-layer execution time and energy consumption values to obtain the execution time and energy consumption of a ResNet-50 inference pass.

IV. Experiment Results

A. Execution Time

Fig. 5 shows the execution time comparison of SPRINT, Simba, and POPSTAR in 21 different ResNet-50 layers and in one complete inference pass. The per-layer execution time values are normalized to the execution time of L21 (fc1000) in Simba while the execution time values for complete inference passes are normalized to the execution time of one complete inference in Simba. As compared to Simba, SPRINT achieves execution time reduction in the range of 31% (L11: res4a_branch1) to 49% (L17: res5a_branch2a). The difference in reduction of execution time comes from (1) the average hops of inter-chiplet communication in Simba, (2) the input channel C and output channel K that determine the utilization rate of PEs in SPRINT, and (3) the IA reuse distance that largely determines the IA broadcast efficiency in SPRINT. As compared to POPSTAR, SPRINT achieves execution time reduction in the range of 6% (L21: fc1000) to 15% (L1: conv1). This indicates the effectiveness of the architecture and dataflow co-design. SPRINT performs better than POPSTAR because (1) SPRINT exploits the ease of broadcast feature better than POPSTAR through package-level data partition, and (2) SPRINT allocates higher bandwidth for communication between the GB and accelerator chiplets. By accumulating the execution time across all layers, we obtain an indication of the execution time of an inference pass (the execution time of non-convolution layers is not included). SPRINT performs a ResNet-50 inference pass 46% and 12% faster than Simba and POPSTAR, respectively.

B. Energy Consumption

Fig. 6 shows the energy consumption comparison of SPRINT, Simba, and POPSTAR in 21 different ResNet-50 layers and in one complete inference pass. The per-layer energy consumption values are normalized to the energy consumption of L21 (fc1000) in Simba while the energy consumption values for complete inference passes are normalized to the energy consumption of one complete inference in Simba. As compared to Simba, SPRINT achieves energy consumption saving in the range of 51% (L11: res4a_branch1) to 69% (L2: res2a_branch2a). This mainly comes from the low energy consumption of inter-chiplet communication in SPRINT. As compared to POPSTAR, SPRINT achieves energy consumption saving in the range of 53% (L11: res4a_branch1) to 54% (L1: conv1). The energy savings observed in different layers are similar because SPRINT photonic inter-chiplet network requires fewer MRRs than the photonic crossbar in POPSTAR. By accumulating the energy consumption across all layers, we observe that SPRINT achieves 61% and 52% energy saving as compared to Simba and POPSTAR, respectively.

C. Scalability

We explore the scalability of SPRINT architecture by varying the number of accelerator chiplets in the system from 4 to 128.

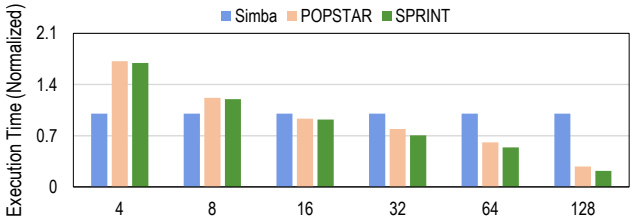

Fig. 7: Execution time comparison when increasing the number of accelerator chiplets, normalized to Simba.

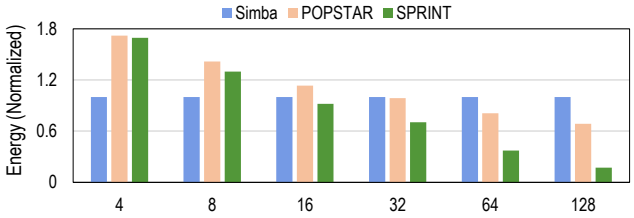

Fig. 8: Energy consumption comparison when increasing the number of accelerator chiplets, normalized to Simba.

Fig. 7 and Fig. 8 are the results of execution time and energy consumption, respectively. We make the following observations: (1) POPSTAR and SPRINT perform worse than Simba when the number of accelerator chiplets is low due to the costly modulation and receiving processes necessary for data exchange between the electrical and optical domains; (2) POPSTAR and SPRINT show good scalability in terms of execution time as the system scales up due to the distance-independent property of photonic interconnects; (3) the energy consumption of POPSTAR, though scales better than Simba, is significantly higher than that of SPRINT due to the large amount of MRRs required in a photonic crossbar; (4) We project good scalability of SPRINT in terms of execution and energy consumption when the system scales beyond 128 accelerator chiplets.

D. Implementation Cost

Micro-bump Area: We assume 36 μm micro-bump pitch size in our work. To achieve 100 $GB/s/chiplet$ bandwidth, there are in total 371 wires connecting the transmitters, receivers and RCU on an accelerator chiplet to corresponding MRRs on the photonic interposer. This only consumes 7.6% of the area of an accelerator chiplet, which is about 6.3 mm^2. As most micro-bumps can be implemented underneath the accelerator chiplet, we assume that they do not incur additional area overhead.

Number of MRRs: To achieve 100 $GB/s/chiplet$ bandwidth, SPRINT requires 14 K MRRs while POPSTAR requires 330 K MRRs. This significant difference in the number of required MRRs results from a fact that the number of MRRs scales linearly and quadratically with the number of accelerator chiplets in SPRINT and POPSTAR, respectively. POPSTAR incurs higher implementation cost but performs worse than SPRINT because the photonic crossbar in POPSTAR does not exploit the unique communication features in DNN inference.

V. Conclusions

This paper presents SPRINT, a chiplet-based architecture with photonic interconnects for DNN inference applications. SPRINT introduces two novel designs: (1) a photonic inter-chiplet network that adapts to the specific communication patterns in DNN inference with minimal implementation cost through reconfiguration, and (2) a customized dataflow that exploits the ease of broadcast and multicast feature of photonics and enables highly parallel DNN computations. Simulation studies prove that SPRINT achieves higher performance, consumes less energy, and exhibits better scalability, as compared to other state-of-the-art chiplet-based architectures.

VI. Acknowledgments

This research was partially supported by NSF grants CCF-1702980, CCF-1812495, CCF-1901165, CCF-1953980, CCF-1513606, CCF-1703013, and CCF-1901192. We sincerely thank the anonymous reviewers for their excellent feedback.

References

[1] K. He, X. Zhang, S. Ren, and J.Sun. Deep Residual Learning for Image Recognition. In *CVPR*, 2016.
[2] C. Szegedy *et al.* Going Deeper with Convolutions. In *CVPR*, 2015.
[3] V. Sze *et al.* Efficient Processing of Deep Neural Networks: A Tutorial and Survey. *Proceedings of the IEEE*, 2017.
[4] R. Mayer and H. Jacobsen. Scalable Deep Learning on Distributed Infrastructures: Challenges, Techniques, and Tools. *ACM CSUR*, 2020.
[5] Y. S. Shao *et al.* Simba: Scaling Deep-Learning Inference with Multi-Chip-Module-Based Architecture. In *MICRO*, 2019.
[6] G. Ascia *et al.* Improving Inference Latency and Energy of DNNs through Wireless Enabled Multi-Chip-Module-Based Architectures and Model Parameters Compression. In *NOCS*, 2020.
[7] R. Hwang *et al.* Centaur: A Chiplet-based, Hybrid Sparse-Dense Accelerator for Personalized Recommendations. *arXiv:2005.05968*, 2020.
[8] Y. Thonnart *et al.* POPSTAR: a Robust Modular Optical NoC Architecture for Chiplet-Based 3D Integrated Systems. In *DATE*, 2020.
[9] P. Grani *et al.* Design and evaluation of awgr-based photonic noc architectures for 2.5 d integrated high performance computing systems. In *HPCA*, 2017.
[10] P. Fotouhi *et al.* Enabling Scalable Chiplet-Based Uniform Memory Architectures with Silicon Photonics. In *MEMSYS*, 2019.
[11] Y. Demir *et al.* Galaxy: A High-Performance Energy-Efficient Multi-Chip Architecture using Photonic Interconnects. In *ICS*, 2014.
[12] A. Narayan *et al.* WAVES: Wavelength Selection for Power-Efficient 2.5 D-Integrated Photonic NoCs. In *DATE*, 2019.
[13] E. Peter, A. Thomas, A. Dhawan, and S. R. Sarangi. Active Microring based Tunable Optical Power Splitters. *Optics Communications*, 2016.
[14] A. Biberman *et al.* Broadband Silicon Photonic Electrooptic Switch for Photonic Interconnection Networks. *IEEE PTL*, 2011.
[15] R. Polster *et al.* Efficiency Optimization of Silicon Photonic Links in 65-nm CMOS and 28-nm FDSOI Technology Nodes. *IEEE TVLSI*, 2016.
[16] A. Parashar *et al.* Timeloop: A Systematic Approach to DNN Accelerator Evaluation. In *ISPASS*, 2019.
[17] P. Rosenfeld, E. Cooper-Balis, and B. Jacob. DRAMSim2: A Cycle Accurate Memory System Simulator. *IEEE CAL*, 2011.
[18] R. Morris, A. Karanth, and A. Louri. Dynamic Reconfiguration of 3D Photonic Networks-on-Chip for Maximizing Performance and Improving Fault Tolerance. In *MICRO*, 2012.
[19] S. Werner *et al.* Designing Low-Power, Low-Latency Networks-on-Chip by Optimally Combining Electrical and Optical Links. In *HPCA*, 2017.
[20] R. Morris and A. Karanth. Power-Efficient and High-Performance Multi-level Hybrid Nanophotonic Interconnect for Multicores. In *NOCS*, 2010.
[21] C. Sun *et al.* DSENT-a Tool Connecting Emerging Photonics with Electronics for Opto-Electronic Networks-on-Chip Modeling. In *NOCS*, 2012.
[22] N. Muralimanohar, R. Balasubramonian, and N. P. Jouppi. CACTI 6.0: A Tool to Model Large Caches. *HP laboratories*, 2009.
[23] A. Stillmaker and B. Baas. Scaling Equations for the Accurate Prediction of CMOS Device Performance from 180 nm to 7 nm. *Integration*, 2017.
[24] A. V. Krishnamoorthy *et al.* Computer Systems based on Silicon Photonic Interconnects. *Proceedings of the IEEE*, 2009.