Flumen: Dynamic Processing in the Photonic Interconnect

Kyle Shiflett
Ohio University
Athens, Ohio, USA
ks117713@ohio.edu

Avinash Karanth
Ohio University
Athens, Ohio, USA
karanth@ohio.edu

Razvan Bunescu
University of North Carolina at Charlotte
Charlotte, North Carolina, USA
rbunescu@uncc.edu

Ahmed Louri
George Washington University
Washington, D.C., USA
louri@gwu.edu

ABSTRACT

In chiplet-based heterogeneous architectures, electrical network-on-package (NoP) designs are typically over-provisioned with routers and channels to provide sufficient bandwidth during periods of high network load. Observing that there are significant periods of low/idle network utilization, prior work has proposed modified network-on-chip (NoC) architectures to enable in-network compute, especially for compute-intensive operations (e.g. linear algebra). However, electrical package-level interconnects impose fundamental energy and bandwidth scaling issues for future chiplet architectures.

This paper proposes Flumen, a dual-purpose photonic interconnect that provides communication at the package-level while also doubling as an accelerator, performing parallel linear computation when network load is low. The proposed architecture utilizes the inherent parallelism of light to create energy-efficient interconnects that support en route computation with minimal changes to the network. By dynamically adjusting the topology, Flumen can change the communication and compute sections of the architecture to adapt to workload fluctuations. Performance evaluation on linear algebra applications shows that Flumen achieves a 2.5× reduction in energy, a 3.6× speedup improvement, and a 9.3× reduction in energy-delay product on average when compared to an electrical mesh network that is used exclusively for communication.

CCS CONCEPTS

• **Hardware → Photonic and optical interconnect**; • **Computer systems organization → Optical computing**; • **Networks → Network on chip**.

KEYWORDS

networks-on-chip, silicon photonics, hardware acceleration

ISCA '23, June 17–21, 2023, Orlando, FL, USA

ACM ISBN 979-8-4007-0095-8/23/06…$15.00
https://doi.org/10.1145/3579371.3589110

ACM Reference Format:
Kyle Shiflett, Avinash Karanth, Razvan Bunescu, and Ahmed Louri. 2023. Flumen: Dynamic Processing in the Photonic Interconnect. In *Proceedings of the 50th Annual International Symposium on Computer Architecture (ISCA '23), June 17–21, 2023, Orlando, FL, USA.* ACM, New York, NY, USA, 13 pages. https://doi.org/10.1145/3579371.3589110

1 INTRODUCTION

The slowing of Moore's law and the continuously increasing demand for processing power has created scaling challenges for future multicore processors [12]. Increasing integrated circuit sizes are faced with decreasing die yields, and as a result higher core counts on a single die are becoming impractical from a cost-per-performance perspective [32]. In addition, scaling performance for modern applications has shifted architectures towards increasingly heterogeneous designs, and the compute demand has surpassed what monolithic integration can provide [40]. Chiplet-based designs arose in response to these challenges, where monolithic processors are disintegrated into several smaller chiplets connected through a shared fabric, such as a silicon interposer [21] or an organic substrate [18].

Modern applications such as deep neural networks (DNNs) place high demand on hardware, with some models requiring billions of multiply-accumulate (MAC) operations [44], and others containing 175 billion trained parameters [4]. Additionally, multimedia processing algorithms such as media encoding have complex data access patterns and multidimensional loop bodies that dominate computation time [34]. Linear algebra operations are at the core of these applications, and although they place high demand on computation resources, they often exhibit low network utilization [39]. Network resources are wasted during idle periods, which could be exploited to relieve pressure from computation cores during periods of high computation demand. Offloading compute tasks to the interconnection network allows computation to occur en route with the data, and moves computation closer to the memory.

Prior works have proposed in-network computation by scheduling operations at the network routers [39] and exploiting dataflow patterns during aggregation [17]. Prior works packetize their operations to move computation through the network, which requires additional decode, buffering, and compute hardware to be added at each network router. These techniques rely on energy-efficient and low latency network-on-chip (NoC) links to move computation between several modified routers, which does not scale well to

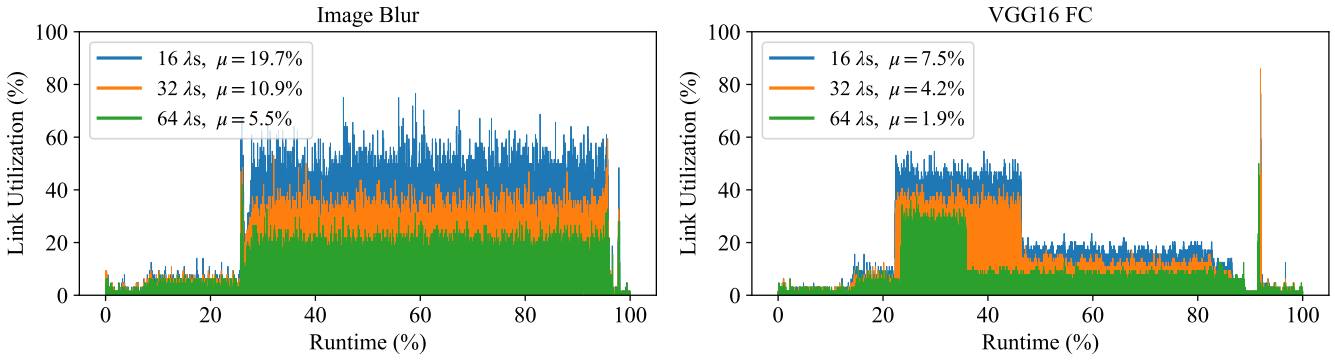


Figure 1: Link utilization and bandwidth sensitivity of a photonic network during Image Blur and VGG16 FC execution.

the package-level interconnects in chiplet systems. Network-on-package (NoP) link energy increases in chiplet-based systems [19], and the packetization of computation breaks the locality otherwise present in multicore cache hierarchies. Data locality is particularly important in linear algebra operations [49, 53], and when NoP link energy surpasses cache access energy, the benefit of electrical in-network computation is diminished.

NoP link distances can be on the order of millimeters to centimeters, and must have high bandwidth and low latency to avoid becoming a system bottleneck [19]. This is a rising concern as designs continue to be split into smaller and more numerous chiplets. Metallic interconnect bandwidth decreases as link lengths increase due to parasitic capacitance, and link power scales linearly with distance [1], posing additional challenges for future large-scale chiplet-based systems. In order to meet the demands of future chiplet systems and efficiently combine communication and computation into a single subsystem, architects must look to emerging technology as an alternative solution.

Silicon photonics can provide the energy-efficient and low-latency package-level interconnects necessary for scaling future chiplet-based systems. Traditionally utilized in communication systems, photonics has emerged as a high-bandwidth energy-efficient alternative for on-chip and off-chip interconnects [1, 48]. Photonic links are favorable for NoP interconnects since they are exempt from the capacitance that afflicts metallic link energy and bandwidth scaling, and are built with low loss waveguides (~1 dB/cm) and energy-efficient modulators (3 fJ/bit) [50]. Light also exhibits additional parallelism not present in electrical interconnects. Several wavelengths can be combined into a single waveguide (~500μm×220μm cross section) without interference using wavelength-division multiplexing (WDM), substantially increasing interconnect bandwidth density. Optical signals can also be easily split for broadcast and multicast communication [30], whereas electrical links require data replication that incurs high energy costs [22].

The intrinsic properties of light also make photonics a potential contender for parallel compute tasks. Photonic accelerators have been proposed to scale DNN inference in terms of energy efficiency and throughput, achieving more than an order of magnitude latency improvement over electronic accelerators [28] and accomplishing throughputs in the range of 11 TOPS [51]. Photonic computation is generally performed in the analog domain using coherent and noncoherent techniques, which occurs as the optical signal propagates through various photonic devices between a transmitter and receiver. Photonic computation can therefore be implemented within the network link itself, rather than in a dedicated compute unit placed in the network router. The benefit of utilizing photonics is two-fold: Photonics provides energy-efficient high-bandwidth NoP links that can double as a computation system, effectively merging two independent domains into a single platform. With fast energy-efficient devices and inherent parallelism of light, photonics may be the scalable solution that combines data movement and computation [29].

This paper proposes Flumen, a dual-function photonic package-level network architecture for combining communication and computation in chiplet-based designs. Flumen's photonic fabric is built using several Mach-Zehnder interferometers (MZIs) connected as one large multiport interferometer called a Mach-Zehnder interferometer mesh (MZIM). Flumen prioritizes communication, and supports point-to-point, physical multicast, and physical broadcast communication patterns. Flumen dynamically accelerates highly-parallel matrix operations using WDM when network resources are available, and supports unitary transformations, general matrix multiplication, and convolution operations. The proposed photonic NoP architecture supports 8-bit equivalent analog computation with minimal changes to the network, allowing seamless transitions between compute acceleration and communication at runtime. When benchmarked on linear algebra applications using contemporary photonic devices, Flumen achieves a 2.5× reduction in energy, a 3.6× speedup improvement, and a 9.3× reduction in energy-delay product on average when compared to a system with an electrical mesh network. The major contributions of the work are as follows:

- **Dual-function photonic interconnect/accelerator:** We propose a Mach-Zehnder interferometer mesh (MZIM) based photonic interconnect with dual functionality that can be reconfigured for data movement during periods of high network load, and acceleration during periods of low network load, both within the same interconnect architecture.
- **Dynamic adaptability:** Flumen can be reconfigured at runtime to adapt the topology to facilitate both communication and computation simultaneously, where separate sections

of the topology implement different functionalities. The network can be partitioned in response to application demands and requirements, thereby improving energy-efficiency and performance.

- **Detailed performance model:** We develop a detailed performance model combining the full-system multicore simulator Sniper [6] and photonic circuit simulator Lumerical INTERCONNECT [27]. Flumen's performance is compared with other network topologies, including electrical ring, electrical mesh, and optical bus, on a variety of benchmarks.

2 MOTIVATION AND BACKGROUND

2.1 Motivation

As mentioned in Section 1, link utilization for linear algebra applications is low, motivating this work. To further illustrate this, the link utilization of several applications using linear algebra operations was recorded during execution. The sensitivity of link utilization for these applications was also considered by underprovisioning link bandwidth for a photonic interconnection network with 16 nodes. Figure 1 shows the link utilization for an image blur application and the fully-connected (FC) layer of the VGG16 convolutional neural network (CNN)[44]. Links use 10 Gbps modulation speed with varying number of wavelengths, so the corresponding link bandwidths are: 16 λs $\Leftrightarrow$ 160 Gbps; 32 λs $\Leftrightarrow$ 320 Gbps; and 64 λs $\Leftrightarrow$ 640 Gbps. In the 64-wavelength high-bandwidth case, average link utilization is only 5.5% and 1.9% for Image Blur and VGG16 FC, respectively. When moving to the 16-wavelength low-bandwidth case, average link utilization is 19.7% and 7.5% for Image Blur and VGG16 FC, respectively. Even with underprovisioned link resources, link utilization is still low on average, leaving ample opportunity for in-network computation.

2.2 Photonic Interconnects

Light is confined and routed on chip using waveguides. Multiple wavelengths of light can propagate in a single waveguide since they do not interfere with one another, and the technique that utilizes this property is wavelength-division multiplexing (WDM). Photonic communication links use WDM to yield high bandwidth interconnects, where each wavelength is modulated independently to carry separate data.

In order to modulate selective wavelengths, resonant devices such as the microring resonator (MRR) are used. MRRs act as (de)multiplexers by resonating at specific wavelengths of light, and the resonant wavelength must be an integer multiple of the effective path length of the MRR: $\lambda_{\text{res}} = \text{n}_{\text{eff}} L/m$, where $m \in \mathbb{Z}^+$ is the integer multiple, n_{eff} is the effective refractive index of the waveguide, and L is the path length of the ring [3]. The resonant wavelength can be modulated using a phase shifter, which alters the effective index of the ring waveguide, and consequently the accumulated phase of a signal propagating through the device.

A photonic communication link utilizing WDM contains several MRRs. At the transmitter there is a modulating MRR for each wavelength to send data down the waveguide, and at the receiver there is an MRR to demultiplex each wavelength for photodetection. The photodiode (PD) outputs a current that is proportional to the incident optical power, which must be amplified up to a usable voltage using a transimpedance amplifier (TIA). Also, MRRs are sensitive to fabrication nonuniformities and temperature, so they require resistive thermal pads to tune the devices to the correct wavelengths. Figure 2 depicts a basic photonic link, where optical power is provided through a fiber from off-chip lasers.

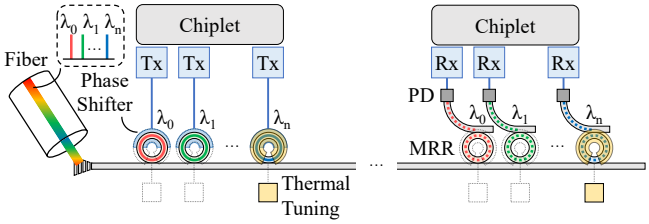


Figure 2: Basic WDM photonic link connecting two chiplets.

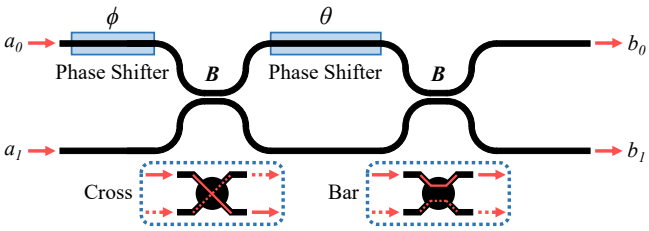


Figure 3: Mach-Zehnder interferometer with phase shifts ϕ and θ, showing the computation $\boldsymbol{b} = \boldsymbol{T}(\theta, \phi)\boldsymbol{a}$, and the cross and bar states.

Non-resonant switching devices, such as the Mach-Zehnder interferometer (MZI) are also useful for modulating optical signals. As opposed to MRRs which selectively modulate certain wavelengths, MZIs perform the same transformation to each wavelength – assuming a broadband response over the range of wavelengths. The MZI is a four-port device that applies an amplitude modulating phase shift $\theta \in [0, \pi]$, and optionally a tuning phase shift $\phi \in [0, 2\pi)$. The MZI is shown in Figure 3, and its transfer matrix is:

$$\boldsymbol{T}(\theta, \phi) = je^{-j\frac{\theta}{2}} \begin{bmatrix} e^{j\phi} \sin\frac{\theta}{2} & \cos\frac{\theta}{2} \\ e^{j\phi} \cos\frac{\theta}{2} & -\sin\frac{\theta}{2} \end{bmatrix} \tag{1}$$

The MZI implements an arbitrary 2×2 unitary transformation on the E-fields of an input vector of optical signals [33]. Two common states of the MZI are the cross state ($\theta = 0$), and the bar state ($\theta = \pi$). In the cross state the top input is switched to the bottom output, and the bottom input is switched to the top output. In the bar state, the top input is switched to the top output, and the bottom input is switched to the bottom output. Any number of intermediate splitting states exist between the cross state and the bar state.

3 FLUMEN ARCHITECTURE

3.1 Photonic Fabric

3.1.1 Mach-Zehnder Interferometer Meshes. The fundamental structure of the Flumen architecture is the Mach-Zehnder interferometer mesh (MZIM), also called a universal multiport interferometer (UMI). The MZIM is a reconfigurable photonic architecture comprised of several layers of MZIs, and is capable of implementing any discrete unitary transformation in the analog domain [33]. An

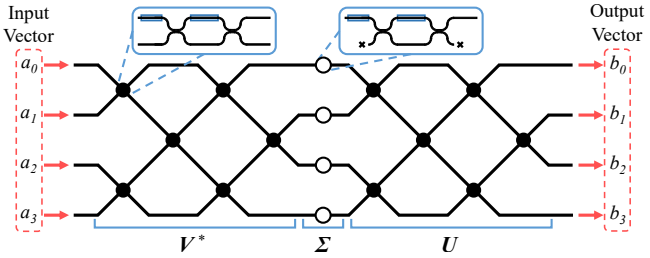


Figure 4: Singular value decomposition MZIM architecture showing the computation $b = U\Sigma V^* a$.

$N \times N$ unitary matrix U is implemented as an N-input MZIM consisting of $N(N-1)/2$ connected MZIs, and the phases θ_i, ϕ_i of each MZI are programmed to implement U [10]. Forward computation occurs as a vector of amplitude-modulated optical inputs propagates through the structure from the N input waveguides to the N output waveguides.

The MZIM architecture can be extended to support non-unitary transformations M through singular value decomposition (SVD): $M = U\Sigma V^*$, where U and V are unitary matrices, Σ is a diagonal matrix of non-negative real numbers called singular values σ_i, and * is the adjoint operator. Figure 4 shows the photonic SVD implementation of a 4×4 matrix. The matrix-vector multiplication $b = Ma = U\Sigma V^* a$ occurs as the optical input vector a propagates from left to right and is transformed into the output vector b. V^* and U are unitary MZIMs connected through a columnn of attenuators Σ. The MZIs used in U and V^* are the same as described in Section 2, but the MZIs used for Σ are only connected at their top two ports, and serve as amplitude modulators rather than tunable beamsplitters. These attenuating MZIs are denoted by open circles in Figure 4. The total number of MZIs in an N-input MZIM SVD architecture is N^2.

The transformation implemented by the MZIM occurs in the analog domain, and the input/output optical signals carry data in their optical power amplitudes and phases. Optical input vectors are modulated using MZIs for computation instead of the MRRs used to modulate data for communication because higher accuracy modulation is needed for computation, and MRR stability is more sensitive to crosstalk and thermal effects. Since the MZIM operates on optical inputs in the analog domain, it must be supported by additional analog electronics to convert between the digital and analog domains. Digital-to-analog converters (DACs) modulate the input signals and implement the MZIM phase shifts. Photodetectors convert the optical signal to an electrical current, TIAs boost this signal up to a usable voltage, and the output voltage is converted using an analog-to-digital converter (ADC).

3.1.2 Flumen Photonic Fabric. MZIMs have not been previously proposed as a network architecture, and Flumen's photonic fabric is a novel variant of the unitary MZIM designed specifically to handle communication and computation simultaneously. Flumen augments an N-input unitary MZIM with a vertical column of N attenuating MZIs, which is shown in Figure 5. By including this additional column of MZIs, the Flumen photonic fabric merges the favorable

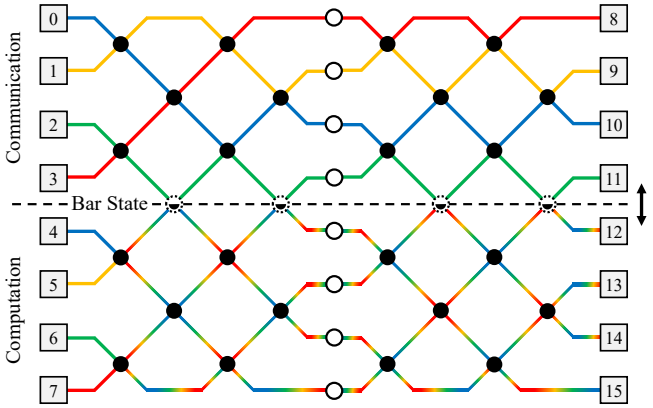


Figure 5: Flumen MZIM architecture, showing dynamic communication/computation partition barrier. Note that the separate colors shown here represent different link paths and are all the same wavelength.

functionality of the unitary MZIM for communication, and the SVD MZIM for computation.

For communication, point-to-point and broadcast/multicast patterns are represented as a unitary matrix, which minimally requires an N-input unitary MZIM. An issue with the basic unitary MZIM for communication is that receivers at a destination node will observe different optical power levels corresponding to the same modulated value, i.e. source-destination paths traverse a differing number of MZIs, therefore experience differing levels of optical loss. Flumen's photonic fabric solves this loss variation issue with its added column of MZIs, which serves to equalize loss differences by attenuating specific source-destination pairs. Take the layout in Figure 5 for example: The longest path to node 15 is 7 MZIs not including the attenuating MZI column, while the shortest path is 4 MZIs.

The dynamic computation functionality of the Flumen MZIM architecture is shown in Figure 5. By placing a row of MZIs into the bar state, they act as reflectors that partition the MZIM into two separate halves. In the top half, point-to-point communication is occurring, and in the bottom half computation is occurring concurrently. Depending on computation or communication demands, this partitioning barrier can be moved to increase computation or communication. By augmenting the 8-input unitary MZIM with attenuating MZIs, when the architecture is partitioned evenly as shown in Figure 5, the resulting partitions are two 4-input SVD MZIMs. The Flumen MZIM architecture supports one large unitary matrix, or several smaller SVD matrices depending on the partition barrier(s). In general, for an N-input Flumen MZIM architecture to be partitioned into two $N/2$-input SVD MZIMs as shown in Figure 5, N must be divisible by 4.

3.2 Communication Mapping

One-to-one and one-to-many communication patterns are easily mapped to Flumen's photonic fabric. For one-to-one communication, links are constructed using MZI cross states ($\theta = 0$) and bar states ($\theta = \pi$). An MZI in the bar state is akin to a reflector, while

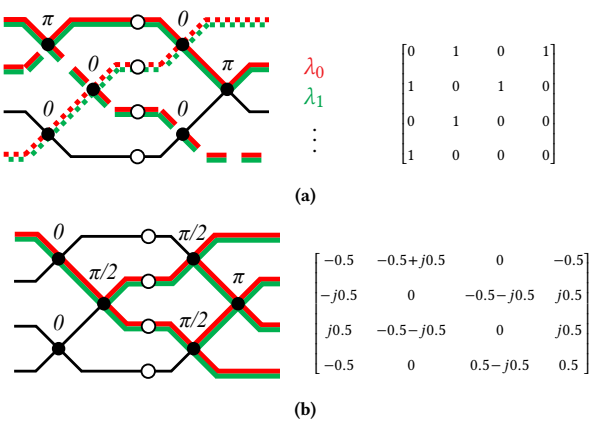


Figure 6: Four-input Flumen MZIM communication showing the θ phase settings and corresponding transfer matrix for: (a) point-to-point communication, and (b) broadcast communication.

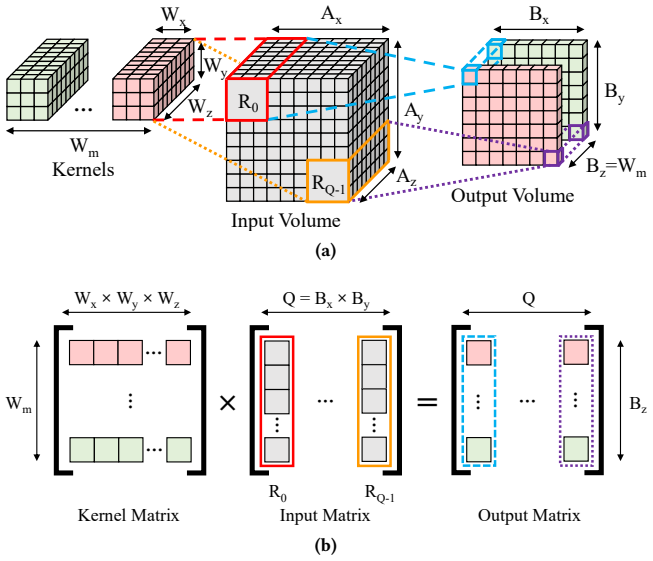


Figure 7: (a) Forward computation of a convolutional layer. (b) Convolutional layer computation reorganized as matrix multiplication.

an MZI in the cross state is viewed as transmissive. The sequence of many reflections and transmissions constructs a non-blocking communication pattern. The MZIM physically resembles a multi-stage interconnection network, however with one-to-one communication it behaves like a crossbar switch. This is because once an optical signal enters the network, it will continue to propagate unimpeded through each MZI until photodetection. One-to-one communication patterns are represented with a unitary adjacency matrix. An example one-to-one mapping with its corresponding adjacency matrix is shown in Figure 6(a). Note that each wavelength is subject to the same MZI transformation as described in 2.2.

One-to-many communication patterns are achieved using intermediate MZI splitting states between the cross and bar states. For example, a 50:50 splitting ratio is achievable when $\theta = \pi/2$ for a single input, which can be used to construct a broadcast tree as shown in Figure 6(b). The unitary matrix corresponding to the broadcast tree in Figure 6(b) is not immediately obvious since these matrices operate on E-fields, but it is more intuitive to think in terms of the optical power amplitudes at the output: $P \propto |E|^2$. If the matrix transformation is performed on the input vector $[1\ 0\ 0\ 0]^T$, then the magnitude of the output vector E-fields are squared, the result is $[0.25\ \ 0.25\ \ 0.25\ \ 0.25]^T$.

3.3 Computation Mapping

3.3.1 Matrix Multiplication Organization. In order for the matrix $\boldsymbol{M}$ to be implemented using an SVD MZIM circuit, it must have singular values $0 \leq \sigma_i \leq 1$ because the optical inputs cannot be amplified at the Σ layer by an arbitrary amount without prior knowledge of these inputs. For energy conservation of the input vector of fields $\boldsymbol{a}$ to be realized, the following condition must be met: $\boldsymbol{b} = \boldsymbol{M}\boldsymbol{a}$, $\boldsymbol{a}^*\boldsymbol{a} \geq \boldsymbol{b}^*\boldsymbol{b} \implies 0 \leq \sigma_i \leq 1$. As a consequence, arbitrary matrices $\boldsymbol{M}$ are not directly implementable in an SVD MZIM, and require a pre-transformation to be performed to guarantee $0 \leq \sigma_i \leq 1$: $\boldsymbol{M}_s = \boldsymbol{M}/\|\boldsymbol{M}\|_2 \implies \sigma_{\max}(\boldsymbol{M}_s) = 1$, where $\|\cdot\|_2$ denotes the spectral norm. The spectral norm of $\boldsymbol{M}$ is equal to its largest singular value. The scaled matrix $\boldsymbol{M}_s$ is then guaranteed to be implementable in a SVD MZIM circuit. To obtain the final transformation output $\boldsymbol{b}$, the result $\boldsymbol{b}_s = \boldsymbol{M}_s\boldsymbol{a}$ must be scaled back by $\|\boldsymbol{M}\|_2$.

In order to implement a matrix $\boldsymbol{M} \in \mathbb{R}^{n\times m}$ in an N-input Flumen MZIM, $\boldsymbol{M}$ must be zero padded along both dimensions to the nearest multiple of N, giving $\widehat{\boldsymbol{M}} \in \mathbb{R}^{\hat{n}\times\hat{m}}$:

$$\hat{\boldsymbol{b}} = \widehat{\boldsymbol{M}}\hat{\boldsymbol{a}}, \quad \widehat{\boldsymbol{M}} = \begin{bmatrix} \boldsymbol{M} & \dots & 0 \\ \vdots & \ddots & 0 \\ 0 & 0 & 0 \end{bmatrix}, \ \hat{\boldsymbol{a}} = \begin{bmatrix} \boldsymbol{a} \\ \vdots \\ 0 \end{bmatrix}, \ \hat{\boldsymbol{b}} = \begin{bmatrix} \boldsymbol{b} \\ \vdots \\ 0 \end{bmatrix} \tag{2}$$

Since the N-input Flumen MZIM implements an $N \times N$ matrix, $\widehat{\boldsymbol{M}}$ must be divided into $(i \times j)$ $N \times N$ sub-blocks, where $i = \hat{n}/N$ and $j = \hat{m}/N$. Computation is then carried out as a block matrix multiplication:

$$\hat{\boldsymbol{b}} = \begin{bmatrix} \widehat{\boldsymbol{M}}_{00} & \dots & \widehat{\boldsymbol{M}}_{0(j-1)} \\ \vdots & \ddots & \vdots \\ \widehat{\boldsymbol{M}}_{(i-1)0} & \dots & \widehat{\boldsymbol{M}}_{(i-1)(j-1)} \end{bmatrix} \begin{bmatrix} \hat{\boldsymbol{a}}_0 \\ \vdots \\ \hat{\boldsymbol{a}}_{(j-1)} \end{bmatrix} \tag{3}$$

Each $N \times N$ block matrix multiplication will generate a set of partial sums. The partial sums from several sub-block multiplications will be accumulated to obtain the final output elements. For example, $\hat{\boldsymbol{b}}_0 = \sum_{k=0}^{j-1} \widehat{\boldsymbol{M}}_{0k}\hat{\boldsymbol{a}}_k$ requires the accumulation of j partial N-element vectors. Computation with an MZIM in this manner means that all multiplications occur in the photonic domain, and each multicore chiplet is responsible only for the accumulation of partials.

The MZIM is an efficient way to rapidly compute matrix products. An $N \times N$ matrix-vector multiplication (MVM) occurs as a single operation in the MZIM, which would otherwise require N^2 multiplication operations and $N(N-1)$ addition operations in the digital domain. The parallelism of photonics can be further exploited to

increase computation density and throughput. By utilizing multiple wavelengths of light similar to a WDM communication link, multiple parallel MVMs can be computed simultaneously. Each input vector $\boldsymbol{a_i}$ to be multiplied by the matrix $\boldsymbol{M}$ is carried on a separate wavelength λ_i. If there are p wavelengths used for computation, the MZIM computes p parallel MVMs, or equivalently computes the matrix-matrix product $\boldsymbol{MA}$, $\boldsymbol{A} \in \mathbb{R}^{N \times p}$ in a single cycle, where $\boldsymbol{A} = [\boldsymbol{a}_0^T\ \boldsymbol{a}_1^T\ \ldots\ \boldsymbol{a}_{p-1}^T]$.

3.3.2 Convolution Organization. The MZIM can also support the convolution operation, which is common in image processing and is the fundamental operation in CNNs. In CNNs, a convolutional layer implements the convolution operation on a set of activations called the input volume, and generates a set of activations called the output volume. The convolution operation, shown in Figure 7(a), is a sliding-window dot product taken between kernels, which hold network weights, and a receptive field in the input volume. Each dot product between a kernel and receptive field produces an element in the output volume, and receptive fields are moved across the entire input volume with stride S.

A convolutional layer is organized as a matrix multiplication using the im2col method [7] for computation in an SVD MZIM circuit, as shown in Figure 7(b). The kernel matrix is comprised of all kernels, where each row of the matrix is a raveled kernel. The kernel matrix has shape $W_m \times (W_x \times W_y \times W_z)$, and left multiplies the input matrix. The input matrix contains all receptive fields, where each column of the matrix is a raveled receptive field. The input matrix has shape $(W_x \times W_y \times W_z) \times Q$, where $Q = B_x \times B_y$. In the SVD MZIM circuit, receptive fields are transmitted on separate wavelengths if multiple wavelengths are used. The output matrix contains the output volume, and has shape $Q \times B_z$. Each column of the output matrix is a $1 \times 1 \times B_z$ slice of the output volume.

3.3.3 MZI Phase Programming. Each $N \times N$ sub-block of $\widehat{\boldsymbol{M}}$ must be mapped to the MZIM phases. In general, MZIM phases are programmed using diagonalization methods that nullify elements in a target matrix. These phase programming algorithms are explored in depth in [10, 15]. In this work, the $\widehat{\boldsymbol{M}}$ matrix phases are assumed to be precomputed using one of the aforementioned programming algorithms. Although matrix phase mapping could be performed at runtime, it is preferred to have these phases precomputed and stored to prevent excessive overhead, especially for repeatedly used matrices. Conversely, communication phases are programmed at runtime since they are easily realized with predefined MZI states.

3.4 Arbitration, Scheduling, and Operation

Flumen's photonic fabric is managed by the MZIM control unit, which is depicted in Figure 8. The MZIM control unit contains several request buffers for communication and computation access to the photonic fabric. Each endpoint is assigned a dedicated buffer for communication, and compute requests are held in a single buffer for each network edge. The MZIM control unit is connected to each network edge through a shared arbitration waveguide. Chiplets communicate with the MZIM control unit on separate wavelengths through the arbitration waveguide. The MZIM control unit also communicates network utilization information back to the chiplets, so cores can make informed decisions regarding whether to send

Algorithm 1 Flumen scheduling process

```
1:  function SchedulerMain
2:      loop                                          ▷ Comm. partition set I
3:          I, A ← Partitioner(I, A)                  ▷ Comp. partition set A
4:          t ← 0
5:          while t < τ do                            ▷ Partition period τ
6:              for each a ∈ A do                     ▷ Comp. partition a
7:                  if done(a) then
8:                      A ← A \ a                     ▷ Remove a from A
9:                      I ← I ∪ a                     ▷ Include a in I
10:                 end if
11:             end for
12:             for each i ∈ I do                     ▷ Comm. partition i
13:                 WavefrontArb(i)                   ▷ Wavefront arbitration on i
14:             end for
15:             t ← t + 1
16:         end while
17:     end loop
18: end function

19: function Partitioner(I, A)
20:     for each a_req ∈ buff_comp do                 ▷ Comp. partition request a_req
21:         β ← ReqBuffUtil(nodes(a_req), ζ)          ▷ Buffer scan depth ζ
22:         if β ≤ η then                             ▷ Buffer utilization threshold η
23:             A ← A ∪ a_req                         ▷ Include a_req in A
24:             I ← I \ a_req                         ▷ Remove a_req from I
25:         end if
26:     end for
27:     return I, A
28: end function
```

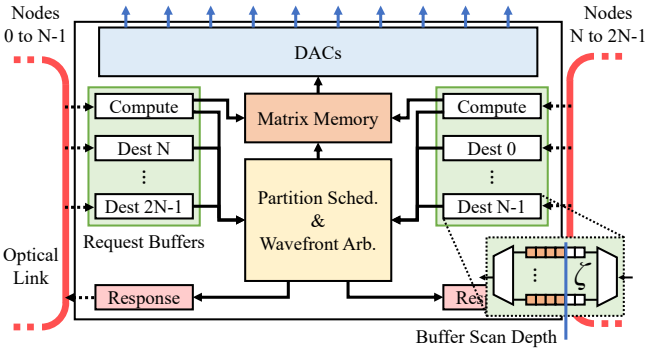


Figure 8: Flumen MZIM control unit.

a compute kernel to the MZIM control unit, or to compute locally. The MZIM control unit contains local matrix memory to hold the precomputed phase mappings for in-network processing. DACs located in the MZIM control unit drive the phase shifters of the MZIs.

The MZIM control unit prioritizes communication over computation, and communication maps are constructed using a wavefront arbiter with additional multicast/broadcast logic. Compute requests are serviced based on the Flumen scheduling algorithm, listed in Algorithm 1. The scheduling algorithm attempts to grant network access to execute compute requests at each network evaluation period τ. If network buffer utilization β is low enough, a computation partition will be created, the controller will notify the requesting node through the arbitration waveguide, and computation will commence. If network buffer utilization β is too high, no compute partition will be created, and the wavefront arbiter will configure

the communication partitions. To prevent excessive compute kernel stalling, nodes will not request compute access if the network utilization conveyed to them by the MZIM control unit is too high, and instead will compute locally.

Network utilization decisions are determined by 2 parameters: the buffer utilization threshold η at a given buffer scan depth ζ. A buffer scan depth ζ was needed because it was observed that a small number of buffers in the MZIM control unit had significantly higher utilization than others for the applications benchmarked (see Section 4.2). This caused high traffic activity among a few nodes to become overlooked by a global buffer utilization parameter, which led to compute partitions excessively blocking communication.

A sensitivity analysis was performed for the algorithm parameters τ, η, and ζ. The partition evaluation period was chosen to be $\tau = 100$ cycles, because this was observed to be the highest average packet latency before network saturation (see Section 5.2). Also, when $\tau > 170$ cycles, a rapid decrease in serviced computation requests was observed, as too many requests were left outstanding. A buffer scan depth of $\zeta = 50\%$ was found to be sufficient for the benchmarked applications, and the buffer utilization threshold was chosen to be $\eta = 40\%$. A value of $\eta \lesssim 30\%$ was too strict, leading to low compute request service, and gave an overall runtime similar to Flumen without in-network processing. It was also observed that $\eta \gtrsim 55\%$ was too aggressive, leading to computation blocking communication, and causing slowdown in some cases.

When a computation kernel completes forward execution, the MZIM control unit configures the computation partition to many-to-one communication pattern, and the MVM results are returned to the requesting node. Once this is complete, the compute partition is deconstructed and made available for communication until the next τ evaluation.

An example Flumen system layout is shown in Figure 9. The photonic MZIM NoP is implemented in the interposer and connects several chiplets. At each chiplet is transceiver hardware that includes the modulators, drivers, DACs, PDs, TIAs, ADCs, serializers, and deserializers for the photonic link. Off-chip lasers provide optical power to the system.

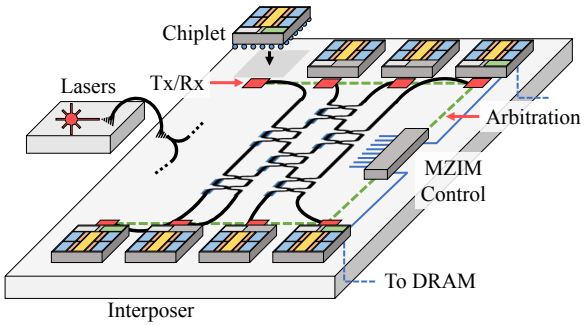


Figure 9: Example 8-chiplet Flumen architecture layout.

Table 1: System-level parameters for performance evaluation.

Component	Parameter	Value
Core	frequency	2.5 GHz
	type	out-of-order
	number	64
	L1i cache	32 kB
	L1d cache	32 kB
L2 (private)	size	512 kB
L3 (shared)	size	16 MB
	concentration	4 cores
Elec. NoP link [37]	energy	1.17 pJ/bit
	bandwidth	800 Gbps
Photonic NoP link	energy (64 λs)	0.703 pJ/bit
	modulation frequency	10 GHz
	bandwidth (64 λs)	640 Gbps
Flumen Compute	computation λs	8
	input modulation freq	5 GHz
	MZIM switch delay	6 ns
	equivalent precision	8 bits

4 EVALUATION METHODOLOGY

4.1 System Setup

The performance of Flumen was evaluated using a 64-core multicore architecture, where each chiplet contains 4 cores with a shared L3 cache. Cores use out-of-order execution with a clock frequency of 2.5 GHz. A list of system parameters is provided in Table 1. The 16-chiplet system is compared using electrical ring (Ring), electrical mesh (Mesh), optical bus (OptBus), and Flumen MZIM interconnection topologies. In an OptBus topology, the network routers are connected by one or more shared circular waveguides, and its variants are commonly explored as photonic interconnection topologies [8, 48]. MZIM architectures similar to Flumen have not been previously proposed as a network architecture, and there is novelty in understanding their benefits for communication alone. There is no compute acceleration equivalent implemented in the Ring, Mesh and OptBus topologies because these computation methods are unique to the structure of the Flumen interconnect, which are based on interferometry of electromagnetic waves – a technique not achievable using metallic interconnects.

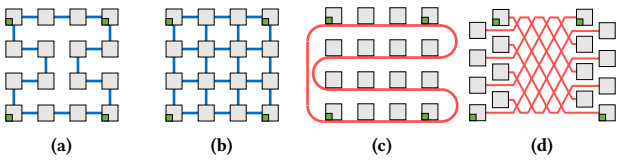


Figure 10: Evaluated NoP topologies: (a) electrical ring, (b) electrical mesh, (c) optical bus, and (d) Flumen MZIM.

Figure 10 shows the layouts of the evaluated topologies. For a fair comparison across network topologies, each NoP was designed to have a similar bisection bandwidth. The bisection bandwidths are: 5.6 Tbps for Ring and Mesh, and 5.1 Tbps for OptBus and Flumen.

Table 2: Photonic and electronic device parameters.

Component	Parameter	Value
Waveguide	straight loss	1.5 dB/cm [9]
	bent loss	3.8 dB/cm [9]
Y-branch	loss	0.3 dB [52]
MRR	radius	5 µm
	thru loss	0.1 dB
	drop loss	1 dB
	modulation power	0.5 mW [54]
	driver power	1 mW [38]
	thermal tuning	1 mW [24]
MZI	phase shifter power	1 nW [46]
	phase shifter loss	0.23 dB [46]
	coupler loss	0.02 dB [26]
Photodiode	sensitivity	20 dBm
	dark current	25 pA [42]
	extinction ratio	7 dB
Laser (off chip)	OWPE	0.2
	RIN	-140 dBc/Hz
ADC	power	29 mW [14]
DAC	power	50 mW [5]
TIA	power	295 µW [36]
Ser. & Deser.	power	1.3 mW [36]

Time taken for network configuration induces a small overhead to Flumen's photonic fabric. Programming MZI phases for communication takes 1 ns [46], which is about 3 processor cycles. Programming an MZIM partition for computation takes longer at 6 ns (15 processor cycles) because the phases for computation need to be more accurate than for communication. These overheads are further quantified in Section 5.

A combination of several modelling tools were used to evaluate system performance. The Sniper multicore full-system simulator [6] was used as the foundation of the simulations, which was extended to include a cycle-accurate network model using Booksim [20]. The photonic circuits were modelled using the Lumerical INTERCONNECT photonic circuit simulator [27]. The photonic device and supporting electronic device parameters used are listed in Table 2, and areas of electronic devices are scaled to 7 nm technology using the scaling equations provided in [45].

The performance of each NoP topology was evaluated on various synthetic traffic patterns using Booksim to characterize how the networks saturate under high network load. Lumerical INTERCONNECT was used to characterize the scaling of photonic circuit losses and latency, which together with the photonic device parameters in Table 2, describes the scaling of communication and computation energy. The Sniper simulator was used to benchmark each architecture, and McPAT [25] was used to obtain energy, runtime, and area results. The results produced by McPAT were scaled to 7 nm technology using equations in [45].

4.2 Benchmarks

The benchmark applications used for architecture evaluation all involve sizable linear algebra operations. Flumen was compared with and without compute acceleration enabled on each of these benchmarks. The benchmarks are detailed below:

- *Image Blur:* The Image Blur application applies a (3×3) Gaussian blur kernel to a (256×256) pixel 24-bit color image. The blurring operation requires approximately 1.7 million multiply-accumulate operations. The Gaussian blur kernel weights are implemented in the MZIM, and receptive field patches are streamed as the optical inputs. The Image Blur application follows the convolution organization shown in Figure 7, which is then decomposed into the sub-block matrices for block matrix multiplication in the MZIM.
- *VGG16 FC:* The VGG16 FC benchmark is the FC-1000 layer in an 8-bit quantized VGG16 CNN. This layer takes as input a 4096-element vector and outputs a 1000-element vector, which is produced through multiplication by a (1000×4096) weight matrix plus a 1000-element bias vector. This neural network layer requires approximately 4.1 million multiply-accumulate operations to compute. The weight matrix is implemented in the MZIM, and the input activations to the layer are sent as the optical inputs.
- *ResNet50 Conv3:* The ResNet50 Conv3 benchmark is one convolutional layer from the conv3_x layer group of an 8-bit quantized ResNet50 model. This layer takes a $(56 \times 56 \times 128)$ volume of activations as input, which is convolved with 128 (3×3) weight kernels. This neural network layer requires approximately 8 million multiply accumulate operations to compute. The convolution layer is organized as shown in Figure 7.
- *JPEG:* This application performs JPEG compression on a (256×384) pixel 24-bit color image, which involves several (8×8) discrete cosine transforms (DCTs). This application requires 1536 2-dimensional DCTs, which involves approximately 1.6 million multiply-accumulate operations.
- *3D Rotation:* This application performs a 3-dimensional graphics rotation on a 306-vertex wire-frame object. Each vertex is represented using a 4-element vector, and the transformation matrix has a shape of (4×4).

5 RESULTS

5.1 Area

The area of each Flumen endpoint is 9.46 mm^2, and 4.2% of this area is for the photonic transceiver. The Flumen 8×8 MZIM plus the MZIM controller occupy 11.2 mm^2, which is 6.9% of the total 162.6 mm^2 occupied by the 64-core architecture. When compared to an electrical mesh architecture that occupies 114.9 mm^2, Flumen's footprint is 17.7 mm^2 larger, which is a 12.2% relative increase. MZIM interconnect area is confined to the interposer, and the area scales well in comparison to the chiplets. An 8×8 MZIM occupies 5.04 mm^2 (~0.5 chiplets in size), and connects 16 chiplets, which have a combined area of 151.36 mm^2. Scaling up to 128 chiplets, a 64×64 Flumen MZIM would occupy 291.20 mm^2 (~16 chiplets

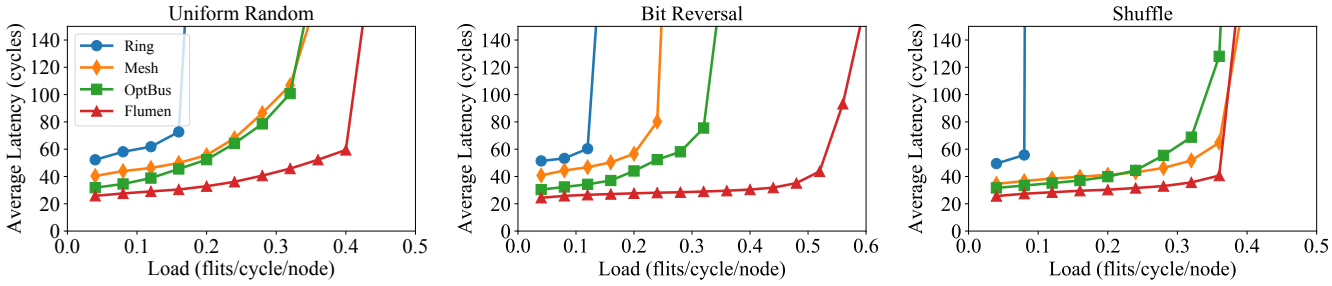
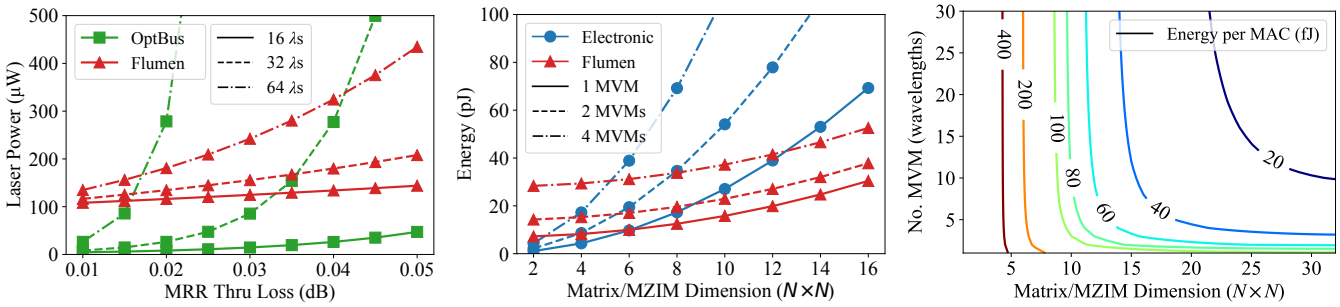


Figure 11: Synthetic traffic evaluation of electrical ring, electrical mesh, optical bus, and Flumen MZIM interconnection topologies.



(a) (b) (c)

Figure 12: (a) Laser power scaling sensitivity to MRR loss for optical bus and Flumen MZIM topologies. (b) Computation energy scaling isolation study between Flumen MZIM and an energy-efficient approximate MAC unit. (c) MAC energy scaling for Flumen photonic computation.

in size), while the 128 chiplets would have a combined area of 1210.88 mm^2.

5.2 Communication System

When benchmarked on synthetic traffic patterns, the Flumen interconnect had the lowest average latency for all offered network loads. Figure 11 shows the latency versus load for uniform random, bit reversal, and shuffle traffic patterns. This shows the benefit of the low-latency photonic links utilized in Flumen interconnect, but also shows the improvement made over the OptBus topology. The OptBus topology performs worse than Flumen because the routers are connected via a shared waveguide and experience higher contention, whereas in Flumen the routers are connected through a non-blocking switching fabric of MZIs. Network energy reduction across the synthetic benchmarks (compared to Ring energy) was 77%, 35%, and 39% for Mesh, OptBus, and Flumen MZIM, respectively. Note that Flumen's average energy was greater than OptBus because Flumen's energy includes the DACs and ADCs required for computation, even though no compute acceleration is occurring. When compared with an MZIM network topology purely for communication, this energy reduces to 28%.

The Flumen interconnect has better energy scaling than an OptBus with an equivalent number of wavelengths. The OptBus is highly sensitive to MRR losses because the worst-case path loss scales proportionally with kp, where k is the number of routers and p is the number of wavelengths. Losses rapidly accumulate as an optical signal propagates past the numerous MRRs attached to the OptBus. The Flumen interconnect loss scales proportional to $k/2 + 2p$. Note that these scaling behaviors are for loss in decibels. Laser power directly depends on the worst-case loss of a photonic interconnect, and can become a significant source of energy consumption in the system. This is illustrated in Figure 12(a), where laser power scaling is evaluated for increasing MRR thru port loss, and increasing wavelengths. Figure 12(a) only shows scaling up to 0.05 dB loss, however the assumed loss used in the architecture evaluation is 0.1 dB. At 32 wavelengths and 0.1 dB MRR thru port loss, laser power is 32.3 mW for OptBus and only 429.6 μW for the Flumen interconnect. This is a 75× laser power reduction compared when compared to OptBus.

5.3 Computation System

The benefits of parallel photonic computation is apparent when comparing the scaling behavior to digital electronics. Flumen energy efficiency was compared to an electrical MAC unit based on a low-power 8-bit approximate multiplier that consumes 0.75 mW at 2.5 GHz [13]. When computing 8×8 matrix multiplication with 4 input vectors, the electrical MAC unit consumed 69.2 pJ and Flumen consumed 33.8 pJ, a 2× improvement. This scales to a 7× reduction in energy for a 8×8 matrix multiplication with 8 input vectors.

Similar scaling behavior is observed for a 16×16 matrix multiplication with 8 input vectors, where the electrical MAC unit consumed 554 pJ and Flumen consumed 82 pJ, which is about a 7× reduction in energy. The compute energy scaling comparison between the electrical MAC unit and Flumen is shown in Figure 12(b). Scaling beyond the 16×16 right limit in Figure 12(b), a large 64×64 Flumen MZIM consumed 0.62 nJ, 1.32 nJ, 2.24 nJ for 1 MVM, 4 MVMs, 8 MVMs, respectively. Compared to the energy-efficient approximate digital circuit, Flumen improved computation energy by 1.8×, 3.4×, and 4.0× for 1 MVM, 4 MVMs, 8 MVMs, respectively.

Flumen computation energy efficiency depends on the MZIM size and number of wavelengths used. During a matrix multiplication computation, there are several sources of static power consumption in the MZIM. This static power includes MRR thermal tuning and MZIM DACs, but the DACs used for the MZI phase shifters constitute the majority of this power. This could be addressed by using multiple phase shifters per DAC with sample and hold circuitry, depending on the sampling rate of the DAC, however this evaluation has assumed one DAC per MZI to give a conservative energy estimate. By increasing the MZIM size and the number of wavelengths, the portion of overall energy consumed by the MZIM control DACs is decreased, and helps to scale energy efficiency per MAC operation for the architecture. The trade-off between MZIM dimension and number of wavelengths for MAC energy is shown in Figure 12(c).

5.4 Benchmark Results

5.4.1 Application Energy. The energy consumption for each network topology was compared to the Flumen interconnect without compute acceleration enabled (Flumen-I), and then with Flumen compute acceleration enabled (Flumen-A). A breakdown of energy consumption is given in Figure 13. Flumen-A improved energy-efficiency by 1.5×, 1.9×, 2.9×, 2.6×, and 4.8× when compared to Mesh topology for Image Blur, VGG16 FC, ResNet50 Conv3, JPEG, and 3D Rotation, respectively, with a geometric mean of 2.5× across all benchmarks. Flumen-A improved energy-efficiency by 1.4×, 1.7×, 2.4×, 2.5×, and 4.2× when compared to Flumen-I topology for Image Blur, VGG16 FC, ResNet50 Conv3, JPEG, and 3D Rotation, respectively, with a geometric mean of 2.3× across all benchmarks.

Flumen-I consumed similar energy as the OptBus system since NoP energy was a small portion of the overall energy, which was observed on all benchmarks. For the Image Blur benchmark, Flumen-I reduced NoP energy consumption by 10.3× and 15.7× compared to Ring and Mesh, respectively. By moving computation into the interconnects, Flumen-A reduced core energy by 2× compared to Ring, and about 1.8× for Mesh, OptBus, and Flumen-I. This led to an increase in NoP energy, however NoP energy is only 3.3% of the overall Flumen-A energy. Flumen-A reduced L1i, L1d, and L2 cache energies, while L3 and DRAM energies did not change significantly. This is because the same data must be fetched from DRAM in all topologies, and the L3 cache is still heavily utilized during compute acceleration for operand and result storage.

Similar behavior was observed in the other benchmark applications. Flumen-A running the 3D Rotation benchmark had the greatest reduction in overall energy, with a 4.7× and 4.8× reduction

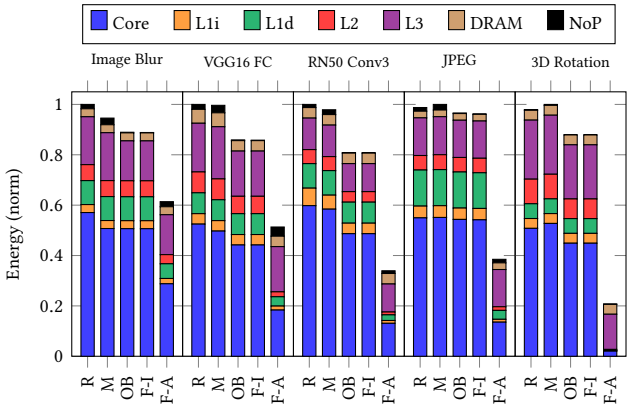


Figure 13: Energy consumption breakdown by component on benchmarks for Ring (R), Mesh (M), OptBus (OB), Flumen-I (F-I), and Flumen-A (F-A).

compared to Ring and Mesh topologies, respectively. 3D Rotation energy reduction was significant because the 4×4 rotation matrix was implemented in two 4-input SVD sub-MZIMs, and the rotation operation did not require the computation cores to accumulate partial sums. The JPEG compression application also had a significant energy reduction when computed using Flumen-A, with an energy reduction of 2.6× over both Ring and Mesh topologies. The 8×8 DCT matrices used in JPEG compression were mapped to the full 8-input unitary MZIM, and also did not require partial sums to be accumulated at the cores. Energy reduction for JPEG compression was not as large as 3D Rotation, but the JPEG algorithm also performed the encoding in the computation cores. Image Blur, VGG16 FC, and ResNet50 Conv3 all had sizable amounts of MZIM partial sums to be accumulated in the cores. ResNet50 Conv3 had the greatest energy reduction of the three benchmarks that involved partial sums because the convolution operation performs more computations per MZIM matrix, reducing the number of MZIM matrix switches due to the shared nature of kernel weights.

5.4.2 Application Speedup. Application speedup of Flumen-A over Ring, Mesh, OptBus, and Flumen-I architectures is shown in Figure 14. The maximum speedup achieved by Flumen-A was 3.3×, 2.0×, 4.5×, 4.0×, and 5.2× for Image Blur, VGG16 FC, ResNet50 Conv3, JPEG, and 3D Rotation, respectively. The average speedup achieved by Flumen-A was 3.1×, 1.9×, 4.1×, 4.0×, and 4.9× for Image Blur, VGG16 FC, ResNet50 Conv3, JPEG, and 3D Rotation, respectively. Flumen-A achieved a speedup of 3.3×, 2.0×, 4.5×, 4.0×, and 5.2× when compared to Mesh topology for Image Blur, VGG16 FC, ResNet50 Conv3, JPEG, and 3D Rotation, respectively, with a geometric mean of 3.6× across all benchmarks. On average across all benchmarks, the phase programming delay plus communication blocking caused about a 9% increase in average packet latency, however the overall application speedup due to compute acceleration justifies this communication overhead.

In general, applications that required fewer partial sum accumulations had higher speedup compared to those with greater partial sum accumulations. Also, applications with higher operand reuse exhibited higher speedup, such as the filters in ResNet50 Conv3,

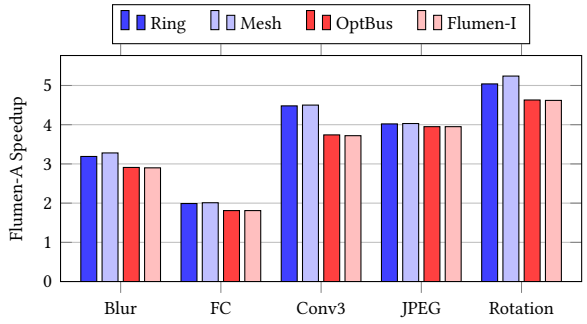


Figure 14: Speedup of Flumen-A over Ring (R), Mesh (M), OptBus (OB), and Flumen-I (F-I) on benchmarked applications.

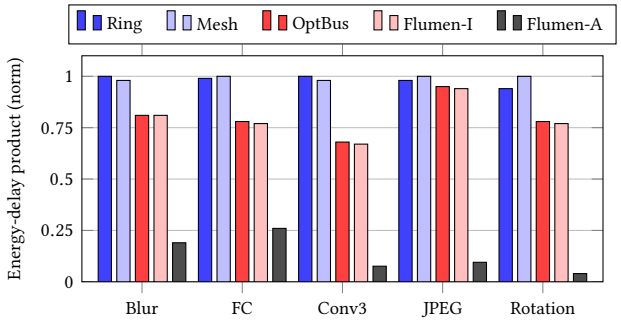


Figure 15: Energy-delay product comparison for Ring (R), Mesh (M), OptBus (OB), Flumen-I (F-I), and Flumen-A (F-A) on evaluated benchmarks.

the blurring kernel in Image Blur, the DCT matrix in JPEG, and the rotation matrix in 3D Rotation. Higher speedup was observed for applications with smaller computation kernels, such as the JPEG and 3D Rotation benchmarks. These factors combined also help identify why the VGG16 FC benchmark had the lowest speedup – it was a large compute kernel with low operand reuse.

5.4.3 Energy-Delay Product. Figure 15 shows the energy-delay product (EDP) of each architecture on the evaluated benchmarks. Flumen-A improved EDP by 5.1×, 3.9×, 13.0×, 10.5×, and 25.2× when compared to Mesh topology for Image Blur, VGG16 FC, ResNet50 Conv3, JPEG, and 3D Rotation, respectively, with a geometric mean of 9.3× across all benchmarks. Flumen-A improved EDP by 4.2×, 3.0×, 8.9×, 9.9×, and 19.5× when compared to Flumen-I topology for Image Blur, VGG16 FC, ResNet50 Conv3, JPEG, and 3D Rotation, respectively, with a geometric mean of 7.4× across all benchmarks.

6 RELATED WORK

The goal of this work was to (a) create a high-bandwidth and energy-efficient photonic network architecture for chiplet-based systems that is competitive with existing interconnects, and (b) identify and repurpose the underutilized network resources to accelerate computation using the parallelism of optics with minimal changes to the network. Prior works have proposed in-network computation by scheduling operations at the network routers [39] and exploiting dataflow patterns during aggregation [17]. These works rely on the energy-efficiency of on-chip interconnects to move computation between network routers, which have been significantly modified to include additional decode logic, operand buffering, and arithmetic cores. Flumen utilizes the same hardware for both communication and computation, and computation occurs within the interconnection links, not in the routers. Prior works also packetize their operands to move between computation routers, which breaks data locality.

Prior works have proposed standalone photonic accelerator chips for DNN inference [2, 28, 35, 43], however, these architectures rely heavily on MRRs to perform their analog computation. MRRs benefit from a small footprint, but crosstalk between MRRs and thermal stability limit the scalability of these designs. MRRs require thermal tuning to stabilize their resonant wavelength, and designs that utilize large numbers of MRRs (135,680 [35], 35,000 [28]) will consume significant energy just for thermal tuning. MZIs do not require thermal tuning like MRRs, and although MZIs occupy a larger footprint, this area overhead is confined to the interposer in Flumen, rather than on chip. Other works have explored the feasibility of MZIMs for neural networks and quantum information processing [16, 41]. ADEPT [11] is a large SVD MZIM ($N = 128$) specifically designed for DNN acceleration. ADEPT provided an 11× inference-per-Joule improvement when compared to an electronic systolic array accelerator. Flumen is an enhanced interconnection network and is not designed to be a standalone accelerator, however [11] demonstrates the potential improvement that MZIMs can provide.

A large group of prior work has explored photonics for on-chip interconnects [8, 23, 31, 47, 48]. These works have explored various topologies, including rings, meshes, crossbars, and multi-buses. These architectures utilize a large number of MRRs, and are susceptible to thermal variations, limiting their scalability. Moreover, prior designs exhibit laser power scalability issues due to the cascaded MRR losses. Flumen improves laser power scalability by using a non-blocking MZI-based interconnect.

7 CONCLUSIONS

This paper proposed Flumen, a dual-purpose photonic interconnect at the package-level that provides communication, while additionally serving as an accelerator during periods of low network load. The proposed architecture utilizes the inherent parallelism of light to construct energy-efficient and scalable interconnects for chiplet-based designs, which support en route computation with minimal changes to the network. By dynamically changing between communication and computation network modes, Flumen is able to adapt to workload fluctuations and provide improved energy-efficiency, speedup, and network resource utilization. When benchmarked on linear algebra applications, the Flumen architecture improved energy-efficiency by 2.5×, achieved a speedup of 3.6×, and reduced EDP by 9.3× on average when compared to an electrical mesh network that is used exclusively for communication.

ACKNOWLEDGMENTS

This research was partially supported by NSF grants CCF-1702980, CCF-1703013, CCF-1812495, CCF-1901165, CCF-1901192, CCF-1936794, and CCF-1953980.

REFERENCES

[1] Theonitsa Alexoudi, Nikolaos Terzenidis, Stelios Pitris, Miltiadis Moralis-Pegios, Pavlos Maniotis, Christos Vagionas, Charoula Mitsolidou, George Mourgias-Alexandris, George T. Kanellos, Amalia Miliou, Konstantinos Vyrsokinos, and Nikos Pleros. 2019. Optics in Computing: From Photonic Network-on-Chip to Chip-to-Chip Interconnects and Disintegrated Architectures. *Journal of Lightwave Technology* 37, 2 (2019), 363–379. https://doi.org/10.1109/JLT.2018.2875995

[2] Viraj Bangari, Bicky A. Marquez, Heidi Miller, Alexander N. Tait, Mitchell A. Nahmias, Thomas Ferreira de Lima, Hsuan-Tung Peng, Paul R. Prucnal, and Bhavin J. Shastri. 2020. Digital Electronics and Analog Photonics for Convolutional Neural Networks (DEAP-CNNs). *IEEE Journal of Selected Topics in Quantum Electronics* 26, 1 (2020), 1–13. https://doi.org/10.1109/JSTQE.2019.2945540

[3] W. Bogaerts, P. De Heyn, T. Van Vaerenbergh, K. De Vos, S. Kumar Selvaraja, T. Claes, P. Dumon, P. Bienstman, D. Van Thourhout, and R. Baets. 2012. Silicon microring resonators. *Laser & Photonics Reviews* 6, 1 (2012), 47–73. https://doi.org/10.1002/lpor.201100017 arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1002/lpor.201100017

[4] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language Models are Few-Shot Learners. In *Advances in Neural Information Processing Systems*, H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin (Eds.), Vol. 33. Curran Associates, Inc., 1877–1901. https://proceedings.neurips.cc/paper/2020/file/1457c0d6bfcb4967418bfb8ac142f64a-Paper.pdf

[5] Pietro Caragiulo, Oscar Elisio Mattia, Amin Arbabian, and Boris Murmann. 2020. A Compact 14 GS/s 8-Bit Switched-Capacitor DAC in 16 nm FinFET CMOS. In *2020 IEEE Symposium on VLSI Circuits*. 1–2. https://doi.org/10.1109/VLSICircuits18222.2020.9162776

[6] Trevor E. Carlson, Wim Heirman, Stijn Eyerman, Ibrahim Hur, and Lieven Eeckhout. 2014. An Evaluation of High-Level Mechanistic Core Models. *ACM Transactions on Architecture and Code Optimization (TACO)*, Article 5 (2014), 23 pages. https://doi.org/10.1145/2629677

[7] Kumar Chellapilla, Sidd Puri, and Patrice Simard. 2006. High Performance Convolutional Neural Networks for Document Processing. In *Tenth International Workshop on Frontiers in Handwriting Recognition*, Guy Lorette (Ed.). Université de Rennes 1, Suvisoft, La Baule (France). https://hal.inria.fr/inria-00112631 http://www.suvisoft.com.

[8] Sai Vineel Reddy Chittamuru, Srinivas Desai, and Sudeep Pasricha. 2017. SWIFT-NoC: A Reconfigurable Silicon-Photonic Network with Multicast-Enabled Channel Sharing for Multicore Architectures. *J. Emerg. Technol. Comput. Syst.* 13, 4, Article 58 (jun 2017), 27 pages. https://doi.org/10.1145/3060517

[9] Lukas Chrostowski, Zeqin Lu, Jonas Flueckiger, Xu Wang, Jackson Klein, Amy Liu, Jaspreet Jhoja, and James Pond. 2016. Design and simulation of silicon photonic schematics and layouts. In *Silicon Photonics and Photonic Integrated Circuits V*, Laurent Vivien, Lorenzo Pavesi, and Stefano Pelli (Eds.), Vol. 9891. International Society for Optics and Photonics, SPIE, 185 – 195. https://doi.org/10.1117/12.2230376

[10] William R. Clements, Peter C. Humphreys, Benjamin J. Metcalf, W. Steven Kolthammer, and Ian A. Walmsley. 2016. Optimal design for universal multiport interferometers. *Optica* 3, 12 (Dec 2016), 1460–1465. https://doi.org/10.1364/OPTICA.3.001460

[11] Cansu Demirkiran, Furkan Eris, Gongyu Wang, Jonathan Elmhurst, Nick Moore, Nicholas C. Harris, Ayon Basumallik, Vijay Janapa Reddi, Ajay Joshi, and Darius Bunandar. 2021. An Electro-Photonic System for Accelerating Deep Neural Networks. https://doi.org/10.48550/ARXIV.2109.01126

[12] Hadi Esmaeilzadeh, Emily Blem, Renée St. Amant, Karthikeyan Sankaralingam, and Doug Burger. 2011. Dark silicon and the end of multicore scaling. In *2011 38th Annual International Symposium on Computer Architecture (ISCA)*. 365–376.

[13] Darjn Esposito, Antonio Giuseppe Maria Strollo, Ettore Napoli, Davide De Caro, and Nicola Petra. 2018. Approximate Multipliers Based on New Approximate Compressors. *IEEE Transactions on Circuits and Systems I: Regular Papers* 65, 12 (2018), 4169–4182. https://doi.org/10.1109/TCSI.2018.2839266

[14] Mingqiang Guo, Jiaji Mao, Sai-Weng Sin, Hegong Wei, and Rui P. Martins. 2020. A 5 GS/s 29 mW Interleaved SAR ADC With 48.5 dB SNDR Using Digital-Mixing Background Timing-Skew Calibration for Direct Sampling Applications. *IEEE Access* 8 (2020), 138944–138954. https://doi.org/10.1109/ACCESS.2020.3012699

[15] Ryan Hamerly, Saumil Bandyopadhyay, and Dirk Englund. 2021. Accurate Self-Configuration of Rectangular Multiport Interferometers. *CoRR* abs/2106.03249 (2021). arXiv:2106.03249 https://arxiv.org/abs/2106.03249

[16] Nicholas C. Harris, Jacques Carolan, Darius Bunandar, Mihika Prabhu, Michael Hochberg, Tom Baehr-Jones, Michael L. Fanto, A. Matthew Smith, Christopher C. Tison, Paul M. Alsing, and Dirk Englund. 2018. Linear programmable nanophotonic processors. *Optica* 5, 12 (Dec 2018), 1623–1631. https://doi.org/10.1364/OPTICA.5.001623

[17] Jiayi Huang, Ramprakash Reddy Puli, Pritam Majumder, Sungkeun Kim, Rahul Boyapati, Ki Hwan Yum, and Eun Jung Kim. 2019. Active-Routing: Compute on the Way for Near-Data Processing. In *2019 IEEE International Symposium on High Performance Computer Architecture (HPCA)*. 674–686. https://doi.org/10.1109/HPCA.2019.00018

[18] Subramanian S. Iyer. 2016. Heterogeneous Integration for Performance and Scaling. *IEEE Transactions on Components, Packaging and Manufacturing Technology* 6, 7 (2016), 973–982. https://doi.org/10.1109/TCPMT.2015.2511626

[19] Natalie Enright Jerger, Ajaykumar Kannan, Zimo Li, and Gabriel H. Loh. 2014. NoC Architectures for Silicon Interposer Systems: Why Pay for more Wires when you Can Get them (from your interposer) for Free?. In *2014 47th Annual IEEE/ACM International Symposium on Microarchitecture*. 458–470. https://doi.org/10.1109/MICRO.2014.61

[20] Nan Jiang, Daniel U. Becker, George Michelogiannakis, James Balfour, Brian Towles, D. E. Shaw, John Kim, and William J. Dally. 2013. A detailed and flexible cycle-accurate Network-on-Chip simulator. In *2013 IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS)*. 86–96. https://doi.org/10.1109/ISPASS.2013.6557149

[21] Ajaykumar Kannan, Natalie Enright Jerger, and Gabriel H. Loh. 2015. Enabling interposer-based disintegration of multi-core processors. In *2015 48th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO)*. 546–558. https://doi.org/10.1145/2830772.2830808

[22] Ammar Karkar, Terrence Mak, Kin-Fai Tong, and Alex Yakovlev. 2016. A Survey of Emerging Interconnects for On-Chip Efficient Multicast and Broadcast in Many-Cores. *IEEE Circuits and Systems Magazine* 16, 1 (2016), 58–72. https://doi.org/10.1109/MCAS.2015.2510199

[23] Cheng Li, Mark Browning, Paul V. Gratz, and Samuel Palermo. 2012. LumiNOC: A power-efficient, high-performance, photonic network-on-chip for future parallel architectures. In *2012 21st International Conference on Parallel Architectures and Compilation Techniques (PACT)*. 421–422.

[24] Guoliang Li, Ashok V. Krishnamoorthy, Ivan Shubin, Jin Yao, Ying Luo, Hiren Thacker, Xuezhe Zheng, Kannan Raj, and John E. Cunningham. 2013. Ring Resonator Modulators in Silicon for Interchip Photonic Links. *IEEE Journal of Selected Topics in Quantum Electronics* 19, 6 (2013), 95–113. https://doi.org/10.1109/JSTQE.2013.2278885

[25] Sheng Li, Jung Ho Ahn, Richard D. Strong, Jay B. Brockman, Dean M. Tullsen, and Norman P. Jouppi. 2009. McPAT: An Integrated Power, Area, and Timing Modeling Framework for Multicore and Manycore Architectures. In *MICRO 42: Proceedings of the 42nd Annual IEEE/ACM International Symposium on Microarchitecture*. 469–480.

[26] Zeqin Lu, Dritan Celo, Patrick Dumais, Eric Bernier, and Lukas Chrostowski. 2015. Comparison of photonic 2×2 3-dB couplers for 220 nm silicon-on-insulator platforms. In *2015 IEEE 12th International Conference on Group IV Photonics (GFP)*. 57–58. https://doi.org/10.1109/Group4.2015.7305944

[27] Lumerical Inc. [n. d.]. https://www.lumerical.com/products/

[28] Armin Mehrabian, Yousra Al-Kabani, Volker J Sorger, and Tarek El-Ghazawi. 2018. PCNNA: A Photonic Convolutional Neural Network Accelerator. In *2018 31st IEEE International System-on-Chip Conference (SOCC)*. 169–173. https://doi.org/10.1109/SOCC.2018.8618542

[29] D. A. B. Miller. 2017. Attojoule Optoelectronics for Low-Energy Information Processing and Communications. *Journal of Lightwave Technology* 35, 3 (2017), 346–396. https://doi.org/10.1109/JLT.2017.2647779

[30] Randy Morris, Evan Jolley, and Avinash Karanth Kodi. 2014. Extending the Performance and Energy-Efficiency of Shared Memory Multicores with Nanophotonic Technology. *IEEE Transactions on Parallel and Distributed Systems* 25, 1 (2014), 83–92. https://doi.org/10.1109/TPDS.2013.26

[31] Randy Morris, Avinash Karanth Kodi, and Ahmed Louri. 2012. Dynamic Reconfiguration of 3D Photonic Networks-on-Chip for Maximizing Performance and Improving Fault Tolerance. In *2012 45th Annual IEEE/ACM International Symposium on Microarchitecture*. 282–293. https://doi.org/10.1109/MICRO.2012.34

[32] Samuel Naffziger, Noah Beck, Thomas Burd, Kevin Lepak, Gabriel H. Loh, Mahesh Subramony, and Sean White. 2021. Pioneering Chiplet Technology and Design for the AMD EPYC™ and Ryzen™ Processor Families : Industrial Product. In *2021 ACM/IEEE 48th Annual International Symposium on Computer Architecture (ISCA)*. 57–70. https://doi.org/10.1109/ISCA52012.2021.00014

[33] Sunil Pai, Ben Bartlett, Olav Solgaard, and David A. B. Miller. 2019. Matrix Optimization on Universal Unitary Photonic Devices. *Phys. Rev. Applied* 11 (Jun 2019), 064044. Issue 6. https://doi.org/10.1103/PhysRevApplied.11.064044

[34] Hyunchul Park, Yongjun Park, and Scott Mahlke. 2009. Polymorphic Pipeline Array: A Flexible Multicore Accelerator with Virtualized Execution for Mobile Multimedia Applications. In *Proceedings of the 42nd Annual IEEE/ACM International Symposium on Microarchitecture* (New York, New York) *(MICRO 42)*. Association for Computing Machinery, New York, NY, USA, 370–380. https://doi.org/10.1145/1669112.1669160

[35] Jiaxin Peng, Yousra Alkabani, Shuai Sun, Volker J. Sorger, and Tarek El-Ghazawi. 2020. DNNARA: A Deep Neural Network Accelerator Using Residue Arithmetic and Integrated Photonics. In *49th International Conference on Parallel Processing - ICPP* (Edmonton, AB, Canada) *(ICPP '20)*. Association for Computing Machinery,

New York, NY, USA, Article 61, 11 pages. https://doi.org/10.1145/3404397.3404467

[36] Robert Polster, Yvain Thonnart, Guillaume Waltener, José-Luis Gonzalez, and Eric Cassan. 2016. Efficiency Optimization of Silicon Photonic Links in 65-nm CMOS and 28-nm FDSOI Technology Nodes. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems* 24, 12 (2016), 3450–3459. https://doi.org/10.1109/TVLSI.2016.2553673

[37] John W. Poulton, John M. Wilson, Walker J. Turner, Brian Zimmer, Xi Chen, Sudhir S. Kudva, Sanquan Song, Stephen G. Tell, Nikola Nedovic, Wenxu Zhao, Sunil R. Sudhakaran, C. Thomas Gray, and William J. Dally. 2019. A 1.17-pJ/b, 25-Gb/s/pin Ground-Referenced Single-Ended Serial Link for Off- and On-Package Communication Using a Process- and Temperature-Adaptive Voltage Regulator. *IEEE Journal of Solid-State Circuits* 54, 1 (2019), 43–54. https://doi.org/10.1109/JSSC.2018.2875092

[38] Michal Rakowski, Julien Ryckaert, Marianna Pantouvaki, Hui Yu, Wim Bogaerts, Kristin de Meyer, Michiel Steyaert, Philippe P. Absil, and Joris Van Campenhout. 2012. Low-Power, 10-Gbps 1.5-Vpp differential CMOS driver for a silicon electro-optic ring modulator. In *Proceedings of the IEEE 2012 Custom Integrated Circuits Conference*. 1–6. https://doi.org/10.1109/CICC.2012.6330643

[39] Karthik Sangaiah, Michael Lui, Ragh Kuttappa, Baris Taskin, and Mark Hempstead. 2020. SnackNoC: Processing in the Communication Layer. In *2020 IEEE International Symposium on High Performance Computer Architecture (HPCA)*. 461–473. https://doi.org/10.1109/HPCA47549.2020.00045

[40] Yakun Sophia Shao, Jason Clemons, Rangharajan Venkatesan, Brian Zimmer, Matthew Fojtik, Nan Jiang, Ben Keller, Alicia Klinefelter, Nathaniel Pinckney, Priyanka Raina, Stephen G. Tell, Yanqing Zhang, William J. Dally, Joel Emer, C. Thomas Gray, Brucek Khailany, and Stephen W. Keckler. 2019. Simba: Scaling Deep-Learning Inference with Multi-Chip-Module-Based Architecture. In *Proceedings of the 52nd Annual IEEE/ACM International Symposium on Microarchitecture* (Columbus, OH, USA) *(MICRO '52)*. Association for Computing Machinery, New York, NY, USA, 14–27. https://doi.org/10.1145/3352460.3358302

[41] Yichen Shen, Nicholas C. Harris, Scott Skirlo, Mihika Prabhu, Tom Baehr-Jones, Michael Hochberg, Xin Sun, Shijie Zhao, Hugo Larochelle, Dirk Englund, and Marin Soljačić. 2017. Deep learning with coherent nanophotonic circuits. *Nature Photonics* 11, 7 (2017), 441–446. https://doi.org/10.1038/nphoton.2017.93

[42] Zhen Sheng, Liu Liu, Joost Brouckaert, Sailing He, and Dries Van Thourhout. 2010. InGaAs PIN photodetectors integrated on silicon-on-insulator waveguides. *Opt. Express* 18, 2 (Jan 2010), 1756–1761. https://doi.org/10.1364/OE.18.001756

[43] Kyle Shiflett, Avinash Karanth, Razvan Bunescu, and Ahmed Louri. 2021. Albireo: Energy-Efficient Acceleration of Convolutional Neural Networks via Silicon Photonics. In *2021 ACM/IEEE 48th Annual International Symposium on Computer Architecture (ISCA)*. 860–873. https://doi.org/10.1109/ISCA52012.2021.00072

[44] Karen Simonyan and Andrew Zisserman. 2015. Very Deep Convolutional Networks for Large-Scale Image Recognition. In *International Conference on Learning Representations (ICLR)*. https://arxiv.org/abs/1409.1556

[45] Aaron Stillmaker and Bevan Baas. 2017. Scaling equations for the accurate prediction of CMOS device performance from 180nm to 7nm. *Integration* 58 (2017), 74–81. https://doi.org/10.1016/j.vlsi.2017.02.002

[46] Mitsuru Takenaka, Jae-Hoon Han, Frédéric Boeuf, Jin-Kwon Park, Qiang Li, Chong Pei Ho, Dongsheng Lyu, Shuhei Ohno, Junichi Fujikata, Shigeki Takahashi, and Shinichi Takagi. 2019. III–V/Si Hybrid MOS Optical Phase Shifter for Si Photonic Integrated Circuits. *Journal of Lightwave Technology* 37, 5 (2019), 1474–1483. https://doi.org/10.1109/JLT.2019.2892752

[47] Scott Van Winkle, Avinash Karanth Kodi, Razvan Bunescu, and Ahmed Louri. 2018. Extending the Power-Efficiency and Performance of Photonic Interconnects for Heterogeneous Multicores with Machine Learning. In *2018 IEEE International Symposium on High Performance Computer Architecture (HPCA)*. 480–491. https://doi.org/10.1109/HPCA.2018.00048

[48] Dana Vantrease, Robert Schreiber, Matteo Monchiero, Moray McLaren, Norman P. Jouppi, Marco Fiorentino, Al Davis, Nathan Binkert, Raymond G. Beausoleil, and Jung Ho Ahn. 2008. Corona: System Implications of Emerging Nanophotonic Technology. In *Proceedings of the 35th Annual International Symposium on Computer Architecture (ISCA '08)*. IEEE Computer Society, USA, 153–164. https://doi.org/10.1109/ISCA.2008.35

[49] Nandita Vijaykumar, Eiman Ebrahimi, Kevin Hsieh, Phillip B. Gibbons, and Onur Mutlu. 2018. The Locality Descriptor: A Holistic Cross-Layer Abstraction to Express Data Locality In GPUs. In *2018 ACM/IEEE 45th Annual International Symposium on Computer Architecture (ISCA)*. 829–842. https://doi.org/10.1109/ISCA.2018.00074

[50] Michael R. Watts, William A. Zortman, Douglas C. Trotter, Ralph W. Young, and Anthony L. Lentine. 2011. Vertical junction silicon microdisk modulators and switches. *Opt. Express* 19, 22 (Oct 2011), 21989–22003. https://doi.org/10.1364/OE.19.021989

[51] Xingyuan Xu, Mengxi Tan, Bill Corcoran, Jiayang Wu, Andreas Boes, Thach G. Nguyen, Sai T. Chu, Brent E. Little, Damien G. Hicks, Roberto Morandotti, Arnan Mitchell, and David J. Moss. 2021. 1 TOPS photonic convolutional accelerator for optical neural networks. *Nature* 589, 7840 (2021), 44–51. https://doi.org/10.1038/s41586-020-03063-0

[52] Yi Zhang, Shuyu Yang, Andy Eu-Jin Lim, Guo-Qiang Lo, Christophe Galland, Tom Baehr-Jones, and Michael Hochberg. 2013. A compact and low loss Y-junction for submicron silicon waveguide. *Opt. Express* 21, 1 (Jan 2013), 1310–1316. https://doi.org/10.1364/OE.21.001310

[53] Zhekai Zhang, Hanrui Wang, Song Han, and William J. Dally. 2020. SpArch: Efficient Architecture for Sparse Matrix Multiplication. In *2020 IEEE International Symposium on High Performance Computer Architecture (HPCA)*. 261–274. https://doi.org/10.1109/HPCA47549.2020.00030

[54] Xuezhe Zheng, Eric Chang, Philip Amberg, Ivan Shubin, Jon Lexau, Frankie Liu, Hiren Thacker, Stevan S. Djordjevic, Shiyun Lin, Ying Luo, Jin Yao, Jin-Hyoung Lee, Kannan Raj, Ron Ho, John E. Cunningham, and Ashok V. Krishnamoorthy. 2014. A high-speed, tunable silicon photonic ring modulator integrated with ultra-efficient active wavelength control. *Opt. Express* 22, 10 (May 2014), 12628–12633. https://doi.org/10.1364/OE.22.012628