An Optical XNOR-Bitcount Based Accelerator for Efficient Inference of Binary Neural Networks

Sairam Sri Vatsavai
Electrical and Computer Engineering
University of Kentucky
Lexington, USA
ssr226@uky.edu

Venkata Sai Praneeth Karempudi
Electrical and Computer Engineering
University of Kentucky
Lexington, USA
kvspraneeth@uky.edu

Ishan Thakkar
Electrical and Computer Engineering
University of Kentucky
Lexington, USA
igthakkar@uky.edu

2023 24th International Symposium on Quality Electronic Design (ISQED) | 979-8-3503-3475-3/23/$31.00 ©2023 IEEE | DOI: 10.1109/ISQED57927.2023.10129294

***Abstract*—Binary Neural Networks (BNNs) are increasingly preferred over full-precision Convolutional Neural Networks (CNNs) to reduce the memory and computational requirements of inference processing with minimal accuracy drop. BNNs convert CNN model parameters to 1-bit precision, allowing inference of BNNs to be processed with simple XNOR and bitcount operations. This makes BNNs amenable to hardware acceleration. Several photonic integrated circuits (PICs) based BNN accelerators have been proposed. Although these accelerators provide remarkably higher throughput and energy efficiency than their electronic counterparts, the utilized XNOR and bitcount circuits in these accelerators need to be further enhanced to improve their area, energy efficiency, and throughput. This paper aims to fulfill this need. For that, we invent a single-MRR-based optical XNOR gate (OXG). Moreover, we present a novel design of bitcount circuit which we refer to as Photo-Charge Accumulator (PCA). We employ multiple OXGs in a cascaded manner using dense wavelength division multiplexing (DWDM) and connect them to the PCA, to forge a novel Optical XNOR-Bitcount based Binary Neural Network Accelerator (OXBNN). Our evaluation for the inference of four modern BNNs indicates that OXBNN provides improvements of up to 62× and 7.6× in frames-per-second (FPS) and FPS/W (energy efficiency), respectively, on geometric mean over two PIC-based BNN accelerators from prior work. We developed a transaction-level, event-driven python-based simulator for evaluation of accelerators (https://github.com/uky-UCAT/B_ONN_SIM).**

I. Introduction

Convolutional Neural Networks (CNNs) have revolutionized the implementation of various artificial intelligence tasks, such as image recognition, language translation, and autonomous driving [1], [2], due to their high inference accuracy. However, the heavy computation and storage requirements of CNNs still limit their application in practice. Therefore, to improve the speed and efficiency of CNN inference, model compression techniques such as quantization are widely employed [3]–[5]. Quantization techniques create compact CNNs compared to their floating-point counterparts by representing the weights/inputs of CNNs with lower precision. The extreme end of the quantization is binarization, i.e., a 1-bit quantization, that allows only two possible values for both inputs and weights, either -1(0) or +1.

Binarization replaces the heavy floating-point vector-dot-product operations (which constitute convolution operations in CNNs) with simple bit-wise XNOR and bitcount operations [6]. Since bit-wise XNOR and bitcount are lightweight operations, binarized CNNs, referred to as binary neural networks (BNNs), provide efficient hardware implementations. Among the BNN hardware implementations from prior works, the silicon-photonic accelerators have shown great promise to provide unparalleled parallelism, ultra-low latency, and high energy efficiency [7], [8]. Prior work [7] utilizes microdisks to realize XNOR-Bitcount processing cores (XPCs) that process the input and weight vectors, whereas [8] uses Microring Resonators (MRRs) in its XPCs to perform XNOR-Bitcount operations. However, these prior works face two shortcomings. First, they use at least two MRRs or microdisks to achieve 1-bit XNOR operation, which increases their area and energy consumption. Second, because of the limited scalability of their XNOR and bitcount circuits, they are forced to decompose the input and weight vectors into a large number of smaller slices before processing them. This generates a large number of partial sums (*psum*s). Accumulating such a large number of *psum*s to obtain the final result, using a *psum* reduction network, can incur a very high latency overhead.

To address these shortcomings, this paper presents a novel Optical XNOR-Bitcount based Binary Neural Network Accelerator (OXBNN). OXBNN employs a novel design of optical XNOR gates (OXGs). Our OXG uses a single MRR to perform a 1-bit XNOR operation, thereby reducing the area and energy consumption compared to prior works. Moreover, OXBNN employs a novel bitcount circuit, referred to as Photo-Charge Accumulator (PCA), which inherently supports the accumulation of a very high number of *psum*s, thereby eliminating the need of using external *psum* reduction networks, to consequently reduce the overall latency and energy consumption of BNN processing.

Our key contributions in this paper are summarized below.

- We present our invented, novel BNN accelerator called OXBNN, which employs an array of single-MRR-based optical XNOR gates (OXGs) and highly scalable bitcount circuits called Photo-Charge Accumulators (PCAs);
- We perform detailed modeling and characterization of our invented OXGs and PCAs using photonics foundry-validated, commercial-grade, photonic-electronic design automation tools (Section III);

- We perform a scalability analysis for our OXBNN and describe a pertinent mapping scheme (Section IV);
- We implement and evaluate OXBNN at the system-level with our in-house simulator (https://github.com/uky-UCAT/B_ONN_SIM), and compare its performance with two well-known photonic BNN accelerators from prior works, for the inferences of four state-of-the-art BNNs (Section V).

II. Preliminaries

A. Binary Neural Networks (BNNs)

BNNs are specific types of CNNs that employ quantization techniques [9] to quantize the weights and inputs to 1-bit values, reducing the storage requirements and computational effort for improved energy efficiency of model inference. With binary quantization, the weights and inputs can only assume two possible values, either -1 or 1 [6], [10]. In general, the *sign* function is the most widely used binary quantization function (Q):

$$Q(x) = sign(x) = x \geq 0\ ?+1 : -1 \tag{1}$$

Like for CNNs [11], a convolution operation for BNNs is also typically decomposed into multiple vector-dot-product (VDP) operations. Each VDP operation of a BNN occurs between two vectors, the individual elements of which are first binarized using Eq. (1). Then, the VDP operation between a binarized weight vector *W* and a binarized input vector *I* can be realized in two steps, in this given order: (i) element-wise (i.e., bit-wise) XNOR of *I* and *W* that produces an XNOR vector; (ii) bitcount of the XNOR vector. This VDP operation is captured in Eq. 2.

$$z = W \odot I = \sum_{i=1}^{S} W_i \odot I_i \tag{2}$$

Here, W_i and I_i, respectively, are the individual bit-elements at index i of the binarized vectors *W* and *I* of size *S* each; $\odot$ denotes the VDP operation (XNOR operation) between binarized vectors *I* and *W* (bit-elements W_i and I_i); $\sum$ represents the bitcount operation.

Using {0,1} instead of {-1,1}: If binary value set {-1,1} is used, obtaining the activation values for the next BNN layer after a convolution operation requires $sign(z)$ for each bitcount result z. On the other hand, if binary value set {0,1} is used, obtaining the activation values for the next BNN layer after a convolution operation requires $compare(z, 0.5\times z_{max})$=$z > 0.5\times z_{max}$?1 : 0 for each bit-count result z, where z_{max} is the size of the binarized vectors *I* and *W*.

B. Processing of BNNs on Hardware

Fig. 1(a) illustrates the convolution between a 3×3 weight channel and a 5×5 input channel. During the convolution, based on the stride parameter, the weight channel slides over the input channel and performs inner products with multiple input channel windows (e.g., four input channel windows are shown in Fig. 1(a) with red, blue, yellow, and green borders),

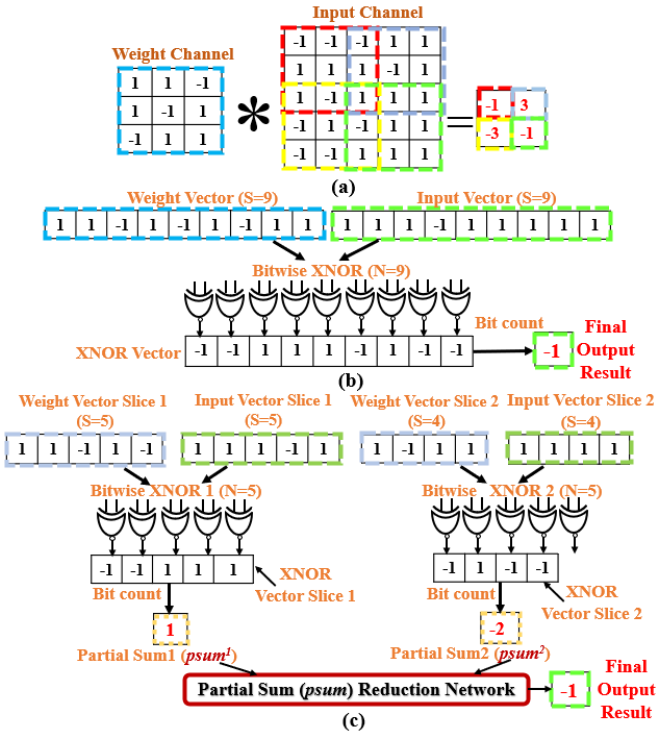

Fig. 1. (a) Illustration of a convolution between a weight and input channel in a Binary Neural Network. Bit-wise XNOR and bitcount operations between a flattened weight vector and input vector, (b) when *S*=*N*=9, and (c) when *N*=5, *S*=9; each input and weight vector of *S*=9 is split into two slices (Slice 1 with *S*=5 and Slice 2 with *S*=4). Binary value set {-1,1} is used in this example.

generating one output value per input channel window. From Fig. 1(b), to perform one such inner product (i.e., corresponding to the input channel window highlighted in green in Fig 1(a)), the input channel window and weight channel are flattened into input and weight vectors of size *S*=9 each. Then, a bitwise XNOR circuit, with a total of *N*=*S*=9 XNOR gates, is employed to generate an XNOR vector. A bitcount circuit then counts the bits in the XNOR vector to evaluate the corresponding inner product output. However, the hardware size $N \neq S$ often. For example, in Fig. 1(c), *S*=9 and *N*=5. In this case, both the input and weight vectors (*S*=9 each) are decomposed into two slices each: Slice 1 with *S*=5 and Slice 2 with *S*=4. These slices are then mapped onto two bitwise XNOR circuits with *N*=5 each, as shown in Fig. 1(c), to consequently produce two XNOR vector slices. Applying bitcount on these XNOR vector slices generates two partial sums (*psums*), i.e., $psum^1$ and $psum^2$. $psum^1$ and $psum^2$ are then sent to a *psum* reduction network to generate the corresponding inner product output. The addition of the *psums* by the *psum* reduction network incurs additional latency and energy overheads while processing BNNs.

C. Related Work on Optical BNN Accelerators

To accelerate CNN inferences with low latency and low energy consumption, prior works proposed various accelerators based on photonic integrated circuits (PICs) (e.g., [12]–[14]).

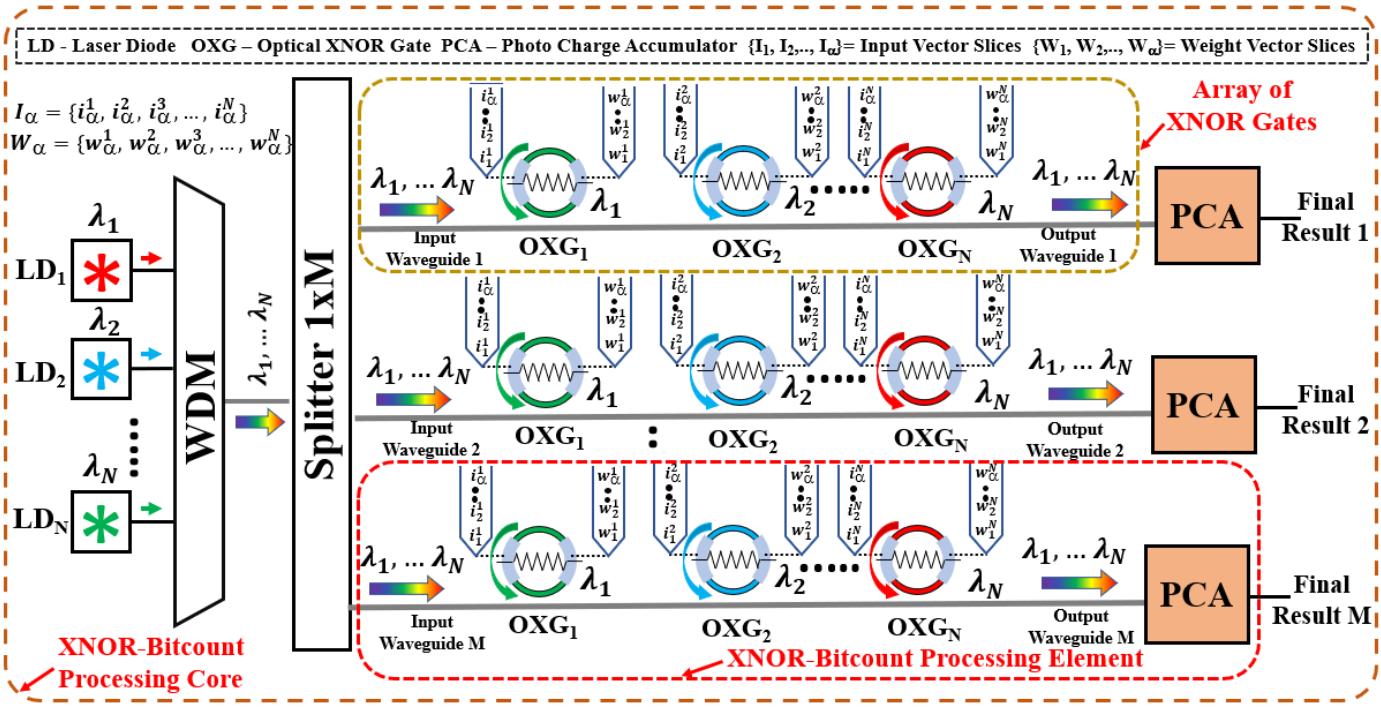

Fig. 2. Schematic of an XNOR-Bitcount Processing Core (XPC) of our OXBNN accelerator. Our OXBNN employs binary value set {0,1}.

These accelerators can be classified as incoherent (e.g., [11]–[13]) or coherent (e.g., [15], [16]). Because of the inherent advantages of incoherent accelerators [13] [17], the BNN-specific incoherent accelerators [8] and [7] were reported. *These optical BNN accelerators from prior works employ binary value set {0,1}*. [8] proposes broadcast and weight styled [18] XNOR-Bitcount circuits, which use heterogeneous MRRs to mitigate fabrication process variations. In contrast, the microdisk-based accelerator [7] proposes an all-optical XNOR-Bitcount circuit that uses optical XNOR gates, optical analog-to-digital converters (ADCs), and PCM-based racetrack memory to enable processing at a very high datarate. However, both [8] and [7] require at least two MRRs or microdisks to perform a 1-bit XNOR operation (in [7], one additional MRR/microdisk is required to modulate the optically applied input operand). Therefore, their XNOR circuits occupy high area and consume high energy. In addition, the bitcount circuits of these prior works can evaluate only one *psum* at a time by counting the bits of one XNOR vector slice at a time. Therefore, these circuits have to store the individual *psum*s temporarily in memory. Once sufficient *psum*s are collected, they can be sent to a *psum* reduction network to produce the final result. Thus, the bitcount circuits from prior works incur high memory footprint for storing *psum*s, and high latency and energy for processing *psum*s. Our OXBNN accelerator addresses these shortcomings of prior works.

III. Our Proposed OXBNN Architecture

A. Overview

The main processing unit of our OXBNN architecture is an XNOR-Bitcount Processing Core (XPC), which is illustrated in Fig. 2. Our XPC has an array of total *N* single-wavelength laser diodes (LDs), with each LD sourcing optical power of $P_{\lambda_i}^{in}$ amount at a distinct wavelength λ_i. The total power from all *N* LDs (at wavelengths λ_1 to λ_N) multiplex into a single photonic waveguide through wavelength division multiplexing (WDM). The optical power containing all these *N* wavelengths is split into *M* input waveguides, each of which connects to an XNOR-Bitcount Processing Element (XPE) (Fig. 2). An XPC contains a total of *M* XPEs.

B. XNOR-Bitcount Processing Element (XPE)

From Fig. 2, an XPE in our OXBNN architecture contains two parts: *(i)* an array of a total of *N* Optical XNOR Gates (OXGs) that generates an XNOR vector (or an XNOR vector slice) containing *N* optical bits, and *(ii)* our invented Photo-Charge Accumulator (PCA) that performs bitcount on the generated XNOR vector (or XNOR vector slice). The value *N* here, which is equal to the number of wavelengths and number of OXGs per XPE, is referred to as the size of the XPE.

1) Array of Optical XNOR Gates (OXGs): In an XPE, an array of a total of *N* OXGs couples to an input waveguide as shown in Fig. 2. Each OXG operates upon a unique wavelength λ_i traversing the input waveguide. Each OXG in the array electrically receives two binary operands (i.e., input bit i_1^N and weight bit w_1^N) from its corresponding drivers (not shown in the figure). The array of OXGs performs a bit-wise logical XNOR between an *N*-bit input vector slice $I_1 = \{i_1^1, i_1^2, .., i_1^N\}$ and an *N*-bit weight vector slice $W_1 = \{w_1^1, w_1^2, .., w_1^N\}$ to produce a resultant *N*-bit XNOR vector slice. Each OXG in the array produces one bit of the resultant XNOR vector slice, and it imprints this bit on its corresponding λ_i (by modulating the optical transmission at λ_i) to be consequently guided to the bitcount circuit (i.e., PCA) via the output waveguide.

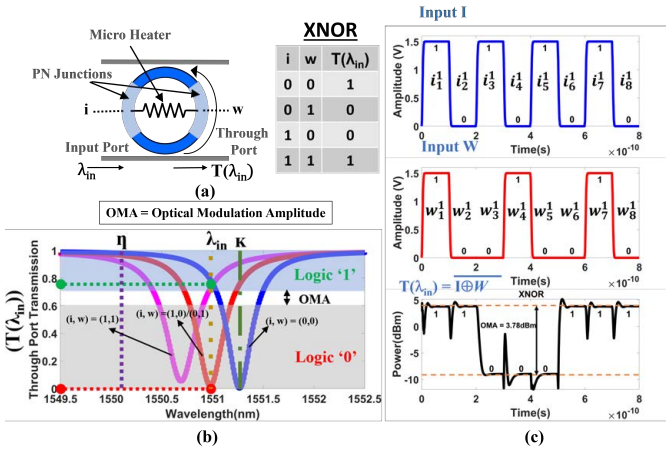

Fig. 3. (a) Schematic of our Optical XNOR Gate (OXG). (b) Spectral operation of OXG. (c) Transient analysis of OXG.

As a result, the PCA receives the N individual optical bits of the N-bit XNOR vector slice concurrently on N distinct wavelengths. The PCA performs bitcount on these optical bits, as explained later. This entire processing step, from the bit-parallel application of the binary input and weight vector slices at the electrical input terminals of the array of N OXGs to the generation of the bitcount result by the PCA, takes very low latency because of the light-speed operation of the XPE. We refer to this processing step mapped on an XPE as a PASS and the corresponding latency as τ. Thus, our XPE can produce one bitcount result for one XNOR vector slice in every single PASS with τ latency. Since τ can be very low (as low as 20 ps), our XPE can achieve very high processing throughput by completing one PASS every τ period. For that, multiple input and weight vector slices $\{I_1, I_2, .., I_\alpha\}$ and $\{W_1, W_2, .., W_\alpha\}$ can be applied to the array of OXGs of an XPE in a serial manner at the predefined data rate (DR) of $\frac{1}{\tau}$. The design and operation of an OXG and PCA are explained next.

Design of an Optical XNOR Gate (OXG): The design of our invented Optical XNOR Gate (OXG) is illustrated in Fig. 3(a). It is an add-drop microring resonator (MRR), which has two operand terminals (realized as embedded PN-junctions) that can take two operand bits i and w as inputs for a predefined time-width (usually a little less than the τ period). Fig. 3(b) shows the passbands of the MRR for different operand inputs and temperature conditions. The MRR's temperature can be increased using the integrated microheater (Fig. 3(a)), to consequently tune its operand-independent resonance from its fabrication-defined initial position η to its programmed position κ (blue passband; Fig. 3(b)), relative to the input optical wavelength position λ_{in}. For each bit combination at the operand terminals ((i,w) = (0,1), (1,0), or (1,1)), the MRR's resonance passband electro-refractively moves to an operand-driven position (red and magenta passbands in Fig. 3(b)). Based on the MRR resonance passband's programmed position κ relative to λ_{in}, the through-port transmission ($T(\lambda_{in})$) of the MRR provides bit-wise logical XNOR operation between the input bits i and w.

To validate the operation of our OXG, we performed the transient analysis, as shown in Fig. 3(c). For that, we modelled and simulated our OXG using the foundry-validated tools from Ansys/Lumerical's DEVICE, CHARGE, and INTERCONNECT suites [19]. Fig. 3(c) shows two input bit-streams $I = \{i_1^1, i_2^1, .., i_8^1\}$ and $W = \{w_1^1, w_2^1, .., w_8^1\}$ applied to the two PN junctions of our OXG at a DR = 10 GS/s. By looking at the output optical trace $T(\lambda_{in})$ in Fig. 3 (c), we can say $T(\lambda_{in}) = \{i_1^1 \odot w_1^1, .., i_8^1 \odot w_8^1\}$, which validates the functionality of our OXG as a logical XNOR gate. From our validation, our OXG has a full passband width at half maximum (FWHM) of 0.35 nm and it can operate at DR of up to 50 GS/s. Our XNOR gate consumes energy of 0.032nJ with an area footprint of 0.011mm^2.

2) Photo-Charge Accumulator (PCA): From Section III-A, the XNOR vector bits generated by an array of OXGs are guided to a PCA circuit, where a bitcount is performed on the XNOR vector bits to generate an output result. Our PCA circuit employs a photodetector and two time integrating receiver (TIR) circuits [20] (one of the TIR1 and TIR2 circuits remains redundant, enabled by the demux and mux; Fig 4). The photodetector generates a current pulse for each optical logic '1' incident upon it. The amplitude of a current pulse generated for an optical logic '0' remains under the noise limit; therefore, a logic '0' remains statistically undetected. The current pulse generated by an optical logic '1' accumulates a certain statistically significant amount of charge on the capacitor of the active TIR circuit (e.g., the circuit with C1 capacitor); as a result, the TIR circuit outputs a detectable analog voltage level [20]. Hence, when more optical '1's are incident upon the photodetector, the total accumulated charge on the active capacitor (e.g., C1), and thus, the accrued output analog voltage level, grows proportionally to the total number of optical '1's that are incident [20]. This is because a current source (a sequence of current pulses) can charge a capacitor linearly following this equation: $\delta V = \frac{i \delta t}{C}$, where i is an incident current pulse, δt is the time-width of the current pulse, C is the capacitance, and δV is the accrued voltage. The final analog voltage accrued at the TIR output, thus, represents the bitcount result (accumulation result) of the incident optical '1's. However, the number of '1's that can be accumulated in such a manner might be limited, as the output of the TIR circuit (Fig. 4) might saturate. Once the output of a TIR circuit saturates, the ongoing accumulation phase ends and the bitcount result (i.e., the final TIR output voltage) is passed through a comparator to generate the activation value for the next BNN layer (as explained in Section II-A). After one accumulation phase, a discharge of the active capacitor (e.g., C1) is needed to prepare the circuit for the next accumulation phase. While capacitor C1 is discharging, the redundant TIR2 circuit with capacitor C2 mitigates the discharge latency by allowing a continuation of a concurrent bitcount.

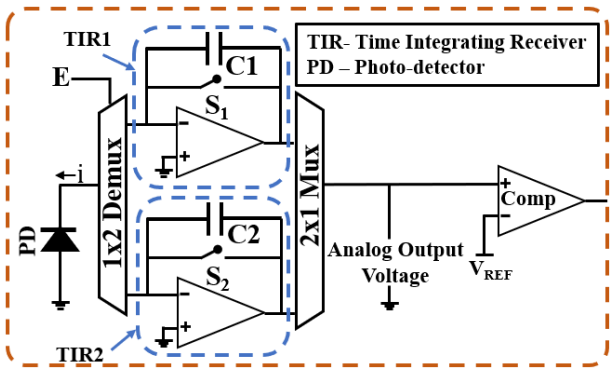

Fig. 4. Photo-Charge Accumulator (PCA) Circuit. V_{REF} is the threshold required in the *compare*() function discussed in Section II-A. Typically, V_{REF} = 2.5V because we consider the dynamic range of TIR to be 5V.

IV. Scalability Analysis and Mapping

A. Scalability of XNOR-Bitcount Processing Cores (XPCs)

To determine the achievable size *N* for our XPC, we adopt scalability analysis equations (Eq. 3, Eq. 4, and Eq. 5) from [21] and [17]. Table I reports the definitions of the parameters and their values used in these equations. We considered Free Spectral Range (FSR=50nm) [21], FWHM=0.35nm (refer Section III-B), and inter-wavelength gap of 0.7nm. For these spectral conditions, we observed minimal crosstalk power penalty for the OXGs operating at DR=50GS/s (<1 dB penalty [22]–[24], which is accounted for as part of parameter $IL_{penalty}$ in the equations (Table I)). Since the XPC of our OXBNN accelerator processes binarized vectors, it requires the bit precision of *B*=1-bit in the equations. We consider *M*=*N* and first solve Eq. 3 and Eq. 4 for a set of DRs={3, 5, 10, 20, 30, 40, 50} GS/s, to find a corresponding set of P_{PD-opt}. Then, we solve Eq. 4 for *N* with the obtained set of P_{PD-opt} values across the set of *DR*s. Table II reports the achievable *N* for our XPC across various *DR*s. As evident, the supported *N* value decreases from *N=66* at 3 GS/s to *N=19* at 50 Gs/s. This achievable *N* value defines the feasible number of OXGs per XPE; thus, this *N* also defines the maximum size of the XNOR vector slice that can be generated in our XPC. Because we consider FSR of 50nm and inter-wavelength gap of 0.7nm, we verify that the maximum *N*=66 can be supported within the FSR (i.e., *N*=66<(FSR/0.7nm)).

$$B = \frac{1}{6.02}\left[20log_{10}\left(\frac{R_s \times P_{PD-opt}}{\beta\sqrt{\frac{DR}{\sqrt{2}}}} - 1.76\right)\right] \quad (3)$$

$$\beta = \sqrt{2q(R_s P_{PD-opt} + I_d) + \frac{4kT}{R_L} + R_s^2 P_{PD-opt}^2 RIN} \quad (4)$$

$$P_{Laser} = \frac{10^{\frac{\eta_{WG}(dB)[N(d_{OXG})+d_{element}]}{10}} M}{\eta_{SMF}\eta_{EC}\eta_{WPE} IL_{i/p-OXG}} \times \frac{P_{PD-opt}}{IL_{penalty}} \times \frac{1}{(OBL_{OXG})^{N-1}(EL_{splitter})^{log_2 M}} \quad (5)$$

TABLE I
Definition and values of various parameters used in Eq. 3, Eq. 4, and Eq. 5 (from [21]) for the scalability analysis. Definitions of PCA parameters γ and α.

Parameter	Definition	Value
P_{Laser}=$P^{in}_{\lambda_i}$	Laser Power Intensity	5 dBm
R_s	PD Responsivity	1.2 A/W
R_L	Load Resistance	50 Ω
I_d	Dark Current	35 nA
T	Absolute Temperature	300 K
RIN	Relative Intensity Noise	-140 dB/Hz
η_{WPE}	Wall Plug Efficiency	0.1
IL_{SMF}	Single Mode Fiber Insertion Loss	0 dB
IL_{EC}	Fiber to Chip Coupling Insertion Loss	1.6 dB
IL_{WG}	Silicon Waveguide Insertion Loss	0.3 dB/mm
$EL_{Splitter}$	Splitter Insertion Loss	0.01 dB
IL_{OXG}	Optical XNOR Gate (OXG) Insertion Loss	4 dB
OBL_{OXG}	Out of Band Loss OXG	0.01 dB
$IL_{penalty}$	Network Penalty	4.8 dB
d_{OXG}	Gap between two adjacent OXGs	20 μm
P_{PD-opt}	Output Photodetector Sensitivity	Table II
α	PCA's Accumulation Capacity (# of XNOR Vectors Slices)	Table II
γ	PCA's Accumulation Capacity (# of accumulated '1's)	Table II

Analysis of PCA's Accumulation Capacity: We modeled the photodetector (PD) of our PCA circuit using the INTERCONNECT tool from Ansys/Lumerical [19] for PD responsivity = 1.2 A/W across different P_{PD-opt} values corresponding to the *N* values in Table II. We extracted the current pulse values generated by the photodetector for the incident optical '1's and '0's corresponding to each P_{PD-opt}. We then imported these values in our MultiSim [25] based model of the PCA with C1=C2=10pF [20], and the TIR gain=50. For these parameters, we simulated the analog output voltage at the PCA's TIR for different bitcount results (i.e., different values of the total number of accumulated '1's). From this analysis, we observed that the maximum number of '1's that can be accumulated by our PCA is limited by the available operating dynamic range of the TIR of our PCA. We considered the TIR's operating dynamic range to be 5V (0V to 5V) and evaluated our PCA's accumulation capacity γ, which we define as the maximum number of '1's that can be accumulated by the PCA within the TIR's operating dynamic range. Our evaluated γ values, for each pair of *N* and corresponding P_{PD-opt}, are reported in Table II. Since our PCA can accumulate a total of γ bits and since each XNOR vector slice in our XPC has a total of *N* bits, our PCA can accumulate a total of α XNOR vector slices, where $\alpha = \frac{\gamma}{N}$. Table II also reports the values of α. As evident, the γ and α values for our PCA can be very large, which provides several substantial benefits as discussed in Section IV-C.

B. Mapping Convolutions on an XPC

As described in Section II-A, for processing a BNN convolution on hardware, both the weight and input channels are flattened into binarized vectors. For mapping of a binary

TABLE II
XPC SIZE *N*, PCA BITCOUNT CAPACITY VALUES (γ AND α), FOR DIFFERENT DATA RATES (DRS).

Datarate (DR) (GS/s)	$P_{PD-opt}(dBm)$	N	γ	α
3	-24.69	66	39682	601
5	-23.49	53	29761	561
10	-21.9	39	19841	508
20	-20.5	29	14880	513
30	-19.5	24	10822	450
40	-18.9	21	9920	472
50	-18.5	19	8503	447

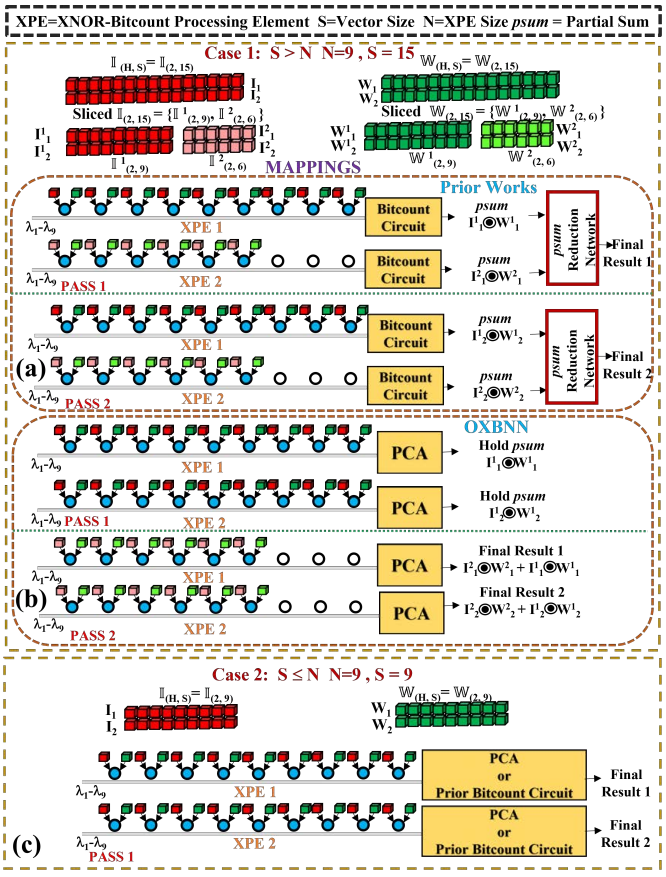

Fig. 5. Example mappings and related operation of our XPC for various cases of the *S* and *N* values. A comparison of our PCA with the bitcount circuit from prior works is also illustrated.

convolution on an XPC (or XPE), these binarized input and weight vectors are represented as matrices. For instance, the input matrix $\mathbb{I}$*(H, S)* has *H* rows corresponding to *H* binarized input vectors of size *S* each. Similarly, the weight matrix $\mathbb{W}$*(H, S)* can also be defined. These matrices $\mathbb{W}$*(H, S)* and $\mathbb{I}$*(H, S)* are mapped onto an XPC containing a total of *M* XPEs of size *N* each. Depending on the relation between *S* and *N*, two cases drive the selection of the appropriate mapping. These cases and their corresponding mappings are illustrated in Fig. 5, for *M*=2, *H*=2, *N*=9, and two distinct values of *S*. These cases are explained below:

Case 1, *S*=15, *S*>*N*, Fig. 5(a) and 5(b): Matrices $\mathbb{I}$ and $\mathbb{W}$ consist of two vectors each, $\{I_1, I_2\}$ and $\{W_1, W_2\}$, respectively. To make the size *S=15* of these vectors $\{I_1, I_2\}$ and $\{W_1, W_2\}$ amenable to the XPE size *N=9*, each of these vectors is split into two slices to yield a set of input vector slices $\{I_1^1, I_1^2, I_2^1, I_2^2\}$ and a set of weight vector slices $\{W_1^1, W_1^2, W_2^1, W_2^2\}$. Since *M=2* is less than the total number of vector slices (i.e., $H \times ceil(S/N)$ = 4), multiple passes are required to complete the processing of these vector slices. Mappings of these vector slices differ between our PCA and the bitcount circuit from prior works [8] and [7], as discussed next.

Mapping for the bitcount circuit from [8] and [7] (Fig. 5(a)): Since M=2, there are two XPEs, namely XPE 1 and XPE 2. During PASS 1 of these XPEs (the definition of a PASS is given in Section III-B), we map $\{I_1^1, W_1^1\}$ onto XPE 1, and $\{I_1^2, W_1^2\}$ onto XPE 2. XPE 1 generates the corresponding XNOR vector, which is accumulated using the bitcount circuit to produce *psum* $I_1^1 \odot W_1^1$. Similarly, XPE 2 generates *psum* $I_1^2 \odot W_1^2$. The generated *psums* are reduced (further accumulated) at the *psum* reduction network, to produce Final Result 1. Similarly, during PASS 2, vector slices $\{I_2^1, I_2^2, W_2^1, W_2^2\}$ are mapped to generate corresponding *psum*s, which are then sent to the *psum* reduction network to produce Final Result 2. Thus, for the bitcount circuits from prior works, there is a need for employing a *psum* reduction network, which leads to a high latency overhead.

Mapping for our OXBNN with PCAs, Fig. 5(b): Our OXBNN maps all the slices of a particular vector to the same XPE. During PASS 1, OXBNN maps $\{I_1^1, W_1^1\}$ to XPE 1, and $\{I_2^1, W_2^1\}$ to XPE 2. XPE 1 charges its PCA's capacitor to generate an analog voltage level that represents *psum* $I_1^1 \odot W_1^1$, whereas XPE 2 charges its PCA's capacitor to generate an analog voltage level that represents *psum* $I_2^1 \odot W_2^1$. Because a PCA can accumulate a total of α vector slices (Section III-B2), the PCAs of XPE 1 and XPE 2 can be made to hold the charge and analog voltage accrued during PASS 1. Then, during PASS 2, XPE 1 and XPE 2 can further grow these held analog voltage levels by the amounts proportional to $I_1^2 \odot W_1^2$ and $I_2^2 \odot W_2^2$, respectively. Thus, at the end of PASS 2, the total accrued analog voltage on the PCA of XPE 1 (XPE 2) would be proportional to $I_1^1 \odot W_1^1 + I_1^2 \odot W_1^2$ ($I_2^1 \odot W_2^1 + I_2^2 \odot W_2^2$). Thus, the PCAs of our OXBNN can accumulate multiple *psums* (a total of α *psums*) inherently. This eliminates the need to employ *psum* reduction networks to consequently yield substantial benefits, as further explained in Section IV-C.

Case 2, *S*=9, *S*≤*N*, Fig. 5 (c): The size *S=9* of the vectors $\{I_1, I_2\}$ and $\{W_1, W_2\}$ matches with the XPE size *N=9*. Thus, in a single pass (PASS 1), our OXBNN maps $\{I_1, W_1\}$ to XPE 1, and $\{I_2, W_2\}$ to XPE 2. XPE 1 and XPE 2 produce Final Result 1 and Final Result 2 corresponding to $I_1 \odot W_1$ and $I_1 \odot W_1$, respectively. In this case, the mapping is identical for our PCA and the bitcount circuits from prior work.

C. Latency and Energy Benefits of PCA

Our PCA provides manifold benefits in terms of both the latency and energy consumption. The latency benefits accrue because our PCA eliminates the need of employing *psum* reduction networks to temporarily store and accumulate *psums*.

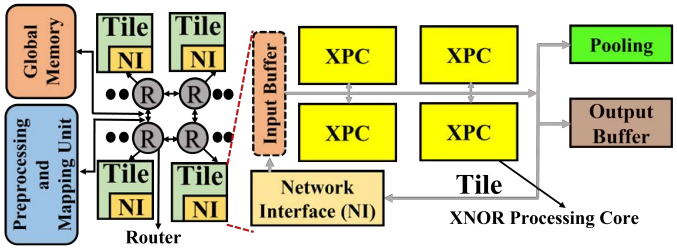

Fig. 6. System-level overview of our OXBNN accelerator.

From Section IV and Table II, our PCA can achieve γ=8503 and α=447 at DR=50 GS/s, which means that our PCA, before it saturates, can accumulate a total of γ=8503 '1's across a total of α=447 XNOR vector slices. As a result, if we operate the OXGs of our OXBNN at DR=50 GS/s, our PCA can inherently accumulate (perform bitcount on) any XNOR vector whose size S is less than γ=8503. Since the maximum XNOR vector size is observed to be S=4608 across all major modern CNNs (e.g., ResNet18, ResNet50, DenseNet121, VGG16, VGG19, GoogleNet, Inception_V3, EffiecientNet_B7, NASNetMobile, MobileNet_V2, and ShuffleNet) [26], our PCA eliminates the need to employ dedicated *psum* reduction networks in our OXBNN accelerator.

V. Evaluation

A. System-Level Implementation of OXBNN.

Fig. 6 illustrates the system-level implementation of our OXBNN accelerator. It consists of global memory that stores BNN parameters and a pre-processing and mapping unit. It has a mesh network of tiles. Each tile contains 4 XPCs interconnected (via H-tree) with an output buffer as well as pooling units.

B. Simulation Setup

For evaluation, we model our OXBNN accelerator from Fig. 6 using our custom-developed, transaction-level, event-driven python-based simulator (https://github.com/uky-UCAT/B_ONN_SIM). We simulated the inference of four BNNs (batch size=1): VGG-small [9], ResNet18 [27], MobileNet_V2 [28], and ShuffleNet_V2 [29]. We binarized all the weights and inputs using the LQ-Nets technique [9]. We evaluate frames-per-second (FPS) and FPS/W (energy efficiency).

We compared our OXBNN with ROBIN [8] and LIGHTBULB [7]. ROBIN and LIGHTBULB operate at different DRs; therefore, we consider two variants of our OXBNN: (1) OXBNN_5 with DR=5GS/s (matching with ROBIN) and *N=53* (Table II), (2) OXBNN_50 with DR=50GS/s (matching with LIGHTBULB) and *N=19* (Table II). We consider two variants of ROBIN: ROBIN Energy-Optimized (ROBIN_EO) and ROBIN Performance-Optimized (ROBIN_PO) [8]. For fair comparison, we perform area proportionate analysis, wherein we altered the XPE count for each photonic BNN accelerator across all of the accelerator's XPCs to match with the area of OXBNN_5 having 100 XPEs. Accordingly, the scaled XPE counts of OXBNN_50 (*N=19*), ROBIN_PO (*N=50*), ROBIN_EO (*N=10*), and LIGHTBULB (*N=16*) are 1123, 183, 916, and 1139, respectively. Table III gives the parameters used for our evaluation.

TABLE III
Accelerator Peripherals and XPE Parameters [17]

	Power(mW)	Latency	Area(mm^2)
Reduction Network	0.050	3.125ns	3.00E-5
Activation Unit	0.52	0.78ns	6.00E-5
IO Interface	140.18	0.78ns	2.44E-2
Pooling Unit	0.4	3.125ns	2.40E-4
eDRAM	41.1	1.56ns	1.66E-1
Bus	7	5 cycles	9.00E-3
Router	42	2 cycles	1.50E-2
EO Tuning	80 μW/FSR	20ns	-
TO Tuning	275 mW/FSR	4μs	-

C. Evaluation Results

Fig. 7(a) compares FPS values (log scale). OXBINN_50 achieves 62×, 8×, and 7× better FPS than ROBIN_EO, ROBIN_PO, and LIGHTBULB, respectively, on gmean across the BNNs. Similarly, OXBNN_5 also outperforms ROBIN_EO, ROBIN_PO, and LIGHTBULB by 54×, 7×, and 16×, respectively, on gmean across the BNNs. OXBNN_5 and OXBNN_50 mainly benefit from the elimination of psum reduction networks. They also benefit from their larger N, compared to the other accelerators. Larger N renders them with higher parallelism and lower α to consequently increase (decrease) their processing throughput (latency).

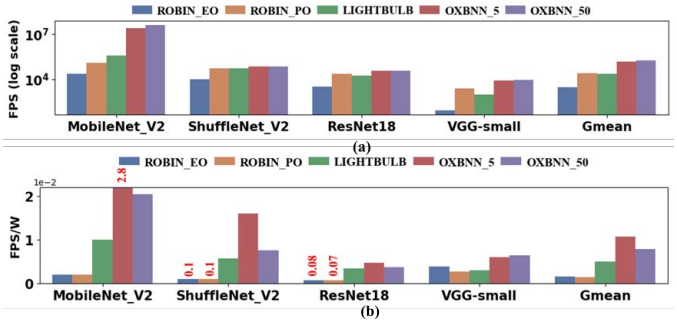

Fig. 7. (a) FPS (log scale) (b) FPS/W for OXBNN versus ROBIN and LIGHTBULB accelerators.

Fig. 7(b) gives the energy efficiency (FPS/W) achieved by each accelerator across various BNNs. Our OXBINN_5 gains 6.8×, 7.6×, and 2.14× better FPS/W than ROBIN_EO, ROBIN_PO, and LIGHTBULB, respectively, on gmean across the BNNs. Similarly, our accelerator OXBNN_50 also outperforms ROBIN_EO, ROBIN_PO, and LIGHTBULB by 4.9×, 5.5×, and 1.5×, respectively, on gmean across the BNNs. The energy benefits of OXBNN_5 and OXBNN_50 are due to the novel OXGs. Due to their single-MRR design, these OXGs consume less energy and static power, compared to the OXGs (containing at least two MRRs or microdisks per OXG) from ROBIN and LIGHTBULB. Moreover, the elimination of the dedicated *psum* reduction network (Section IV-C) also eliminates related high energy consumption. Thus, these benefits collectively render better FPS/W for OXBNN_5 and OXBNN_50.

VI. Conclusions

In this paper, we present a single-MRR-based optical XNOR gate (OXG) and a novel bitcount circuit Photo-Charge Accumulator (PCA). We employ OXGs and PCAs to forge a novel accelerator, called OXBNN, to process the inferences of BNNs. We performed a comprehensive analysis to show the throughput and energy efficiency advantages of OXBNN. Our evaluation results show that OXBNN provides improvements of up to 62× and 7.6× in throughput (FPS) and energy efficiency (FPS/W), respectively, on geometric mean over two state-of-the-art photonic BNN accelerators from prior works.

Acknowledgments

We thank the anonymous reviewers whose valuable feedback helped us improve this paper. We would also like to acknowledge the National Science Foundation (NSF) as this research was supported by NSF under grant CNS-2139167.

References

[1] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *nature*, vol. 521, no. 7553, p. 436, 2015.

[2] S. Dong, P. Wang, and K. Abbas, "A survey on deep learning and its applications," *Comput. Sci. Rev.*, vol. 40, no. C, may 2021. [Online]. Available: https://doi.org/10.1016/j.cosrev.2021.100379

[3] S. Gupta, A. Agrawal, K. Gopalakrishnan, and P. Narayanan, "Deep learning with limited numerical precision," in *Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37*, ser. ICML'15. JMLR.org, 2015, p. 1737–1746.

[4] F. Zhu, R. Gong, F. Yu, X. Liu, Y. Wang, Z. Li, X. Yang, and J. Yan, "Towards unified int8 training for convolutional neural network," in *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 1966–1976.

[5] Y. Gong, L. Liu, M. Yang, and L. D. Bourdev, "Compressing deep convolutional networks using vector quantization," *ArXiv*, vol. abs/1412.6115, 2014.

[6] M. Rastegari, V. Ordonez, J. Redmon, and A. Farhadi, "Xnor-net: Imagenet classification using binary convolutional neural networks," in *European conference on computer vision*. Springer, 2016, pp. 525–542.

[7] F. Zokaee, Q. Lou, N. Youngblood, W. Liu, Y. Xie, and L. Jiang, "Lightbulb: A photonic-nonvolatile-memory-based accelerator for binarized convolutional neural networks," in *2020 Design, Automation Test in Europe Conference Exhibition (DATE)*, 2020, pp. 1438–1443.

[8] F. P. Sunny, A. Mirza, M. Nikdast, and S. Pasricha, "Robin: A robust optical binary neural network accelerator," *ACM Trans. Embed. Comput. Syst.*, vol. 20, no. 5s, sep 2021. [Online]. Available: https://doi.org/10.1145/3476988

[9] D. Zhang, J. Yang, D. Ye, and G. Hua, "Lq-nets: Learned quantization for highly accurate and compact deep neural networks," in *Computer Vision – ECCV 2018: 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part VIII*. Berlin, Heidelberg: Springer-Verlag, 2018, p. 373–390.

[10] M. Courbariaux, I. Hubara, D. Soudry, R. El-Yaniv, and Y. Bengio, "Binarized neural networks: Training deep neural networks with weights and activations constrained to+ 1 or-1," *arXiv preprint arXiv:1602.02830*, 2016.

[11] V. Bangari, B. A. Marquez, H. Miller, A. N. Tait, M. A. Nahmias, T. F. de Lima, H.-T. Peng, P. R. Prucnal, and B. J. Shastri, "Digital electronics and analog photonics for convolutional neural networks (deap-cnns)," *IEEE Journal of Selected Topics in Quantum Electronics*, vol. 26, no. 1, pp. 1–13, 2020.

[12] W. Liu, W. Liu, Y. Ye, Q. Lou, Y. Xie, and L. Jiang, "Holylight: A nanophotonic accelerator for deep learning in data centers," in *2019 Design, Automation Test in Europe Conference Exhibition (DATE)*, 2019, pp. 1483–1488.

[13] F. Sunny, A. Mirza, M. Nikdast, and S. Pasricha, "Crosslight: A cross-layer optimized silicon photonic neural network accelerator," in *2021 58th ACM/IEEE Design Automation Conference (DAC)*, 2021, pp. 1069–1074.

[14] H. Zhang, M. Gu, X. Jiang, J. Thompson, H. Cai, S. Paesani, R. Santagati, A. Laing, Y. Zhang, M. Yung, Y. Shi, F. Muhammad, G. Lo, X. Luo, B. Dong, D. Kwong, L. Kwek, and A.-q. Liu, "An optical neural chip for implementing complex-valued neural network," *Nature Communications*, vol. 12, 01 2021.

[15] C. Demirkiran, F. Eris, G. Wang, J. Elmhurst, N. Moore, N. C. Harris, A. Basumallik, V. J. Reddi, A. M. Joshi, and D. Bunandar, "An electro-photonic system for accelerating deep neural networks," *ArXiv*, vol. abs/2109.01126, 2021.

[16] X. Lin, Y. Rivenson, N. T. Yardimci, M. Veli, Y. Luo, M. Jarrahi, and A. Ozcan, "All-optical machine learning using diffractive deep neural networks," *Science*, vol. 361, no. 6406, pp. 1004–1008, 2018. [Online]. Available: https://www.science.org/doi/abs/10.1126/science.aat8084

[17] S. Sri Vatsavai and I. G. Thakkar, "Photonic reconfigurable accelerators for efficient inference of cnns with mixed-sized tensors," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 41, no. 11, pp. 4337–4348, 2022.

[18] Q. Cheng, J. Kwon, M. Glick, M. Bahadori, L. P. Carloni, and K. Bergman, "Silicon photonics codesign for deep learning," *Proceedings of the IEEE*, vol. 108, no. 8, pp. 1261–1282, 2020.

[19] "Pic design and simulation software - lumerical interconnect," Apr 2021. [Online]. Available: https://www.lumerical.com/products/interconnect/

[20] A. Sludds, S. Bandyopadhyay, Z. Chen, Z. Zhong, J. Cochrane, L. Bernstein, D. Bunandar, P. B. Dixon, S. A. Hamilton, M. Streshinsky, A. Novack, T. Baehr-Jones, M. Hochberg, M. Ghobadi, R. Hamerly, and D. Englund, "Delocalized photonic deep learning on the internet’s edge," *Science*, vol. 378, no. 6617, pp. 270–276, 2022.

[21] M. A. Al-Qadasi, L. Chrostowski, B. J. Shastri, and S. Shekhar, "Scaling up silicon photonic-based accelerators: Challenges and opportunities," *APL Photonics*, vol. 7, no. 2, p. 020902, 2022. [Online]. Available: https://doi.org/10.1063/5.0070992

[22] M. Bahadori, S. Rumley, H. Jayatilleka, K. Murray, N. A. F. Jaeger, L. Chrostowski, S. Shekhar, and K. Bergman, "Crosstalk penalty in microring-based silicon photonic interconnect systems," *Journal of Lightwave Technology*, vol. 34, no. 17, pp. 4043–4052, 2016.

[23] V. S. P. Karempudi, S. Sri Vatsavayi, and I. Thakkar, "Redesigning photonic interconnects with silicon-on-sapphire device platform for ultra-low-energy on-chip communication," in *Proceedings of the 2020 on Great Lakes Symposium on VLSI*, ser. GLSVLSI '20. New York, NY, USA: Association for Computing Machinery, 2020, p. 247–252. [Online]. Available: https://doi.org/10.1145/3386263.3406929

[24] S. S. Vatsavai, V. S. P. Karempudi, and I. Thakkar, "Proteus: Rule-based self-adaptation in photonic nocs for loss-aware co-management of laser power and performance," in *2020 14th IEEE/ACM International Symposium on Networks-on-Chip (NOCS)*. IEEE, 2020, pp. 1–8.

[25] "Multisim." [Online]. Available: https://www.ni.com/en-us/shop/software/products/multisim.html

[26] F. Chollet *et al.*, "Keras," https://keras.io/api/applications/, 2015.

[27] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," 06 2016, pp. 770–778.

[28] A. Howard, A. Zhmoginov, L.-C. Chen, M. Sandler, and M. Zhu, "Inverted residuals and linear bottlenecks: Mobile networks for classification, detection and segmentation," in *CVPR*, 2018.

[29] X. Zhang, X. Zhou, M. Lin, and J. Sun, "Shufflenet: An extremely efficient convolutional neural network for mobile devices," 06 2018, pp. 6848–6856.