LightMat: A Hybrid Photonic-Electronic Computing System for Matrix Multiplication

Hailong Gong[1], Haibo Zhang[2(✉)], Amanda S Barnard[1], Matt Woolley[3], and Rajkumar Buyya[4]

[1] Australian National University, Australia
{hailong.gong, amanda.s.barnard}@anu.edu.au
[2] University of New South Wales, Sydney, Australia
haibo.zhang@unsw.edu.au
[3] University of New South Wales, Canberra, Australia
m.woolley@unsw.edu.au
[4] The University of Melbourne, Australia
rbuyya@unimelb.edu.au

Abstract. Matrix multiplication is the dominant computational operation in AI workloads, where its performance on electronic computing systems is increasingly constrained by memory bandwidth and energy consumption. Photonic computing offers a promising alternative to address these challenges, but existing systems lack support for programmable and reconfigurable matrix computation. We present **LightMat**, a hybrid photonic–electronic computing system designed to accelerate tile-based matrix multiplication by integrating high-speed optical modulation with programmable digital control. LightMat combines a data stream synchronization mechanism with tiled execution and reconstruction. We evaluate a single-core FPGA-controlled photonic prototype and a 32-group, 1536-lane extension in simulation. Across 500 triggers, the prototype shows timing-offset standard deviations below 2 samples and achieves $\sim 2.8\%$ mean relative error on unsigned 8-bit positive integers. In simulation, LightMat achieves over $7\times$ the energy efficiency of Lightning and $1.4\times$ that of an NVIDIA A100 at an assumed 250 W for 512×512 matrices.

Keywords: Photonic Computing · Matrix Multiplication Acceleration · Energy-efficient · High-Performance Systems

1 Introduction

The rapid advancement of artificial intelligence (AI) has driven the development of increasingly large deep learning models, whose computational workloads are dominated by matrix operations such as dense layers, convolutions, and attention mechanisms. As model size scales, the cost of matrix multiplication grows significantly, resulting in a rising computational demand that challenges the performance and energy efficiency of conventional electronic hardware. These limitations reflect the broader challenges of the post-Moore era, where the slowdown of Moore's Law and the end of Dennard scaling restrict further transistor

miniaturization while power density continues to rise. Modern supercomputing systems already operate at tens of megawatts [7], highlighting the unsustainable energy trajectory of electronic computing. These trends highlight the need for new computing paradigms that can meet the growing demands of AI workloads while substantially improving energy efficiency for sustainable AI development.

Photonic computing has emerged as a promising approach for overcoming the performance and energy bottlenecks by leveraging the high bandwidth, low latency, and inherent parallelism of light. Through optical modulation, interference, and detection, photonic systems can perform multiplication and addition operations directly in the optical domain. Data movement over photonic interconnect also consumes substantially less energy than in electrical interconnects. Moreover, optical channels naturally support multi-wavelength parallelism via wavelength-division multiplexing (WDM), allowing multiple operations to be executed simultaneously in a single computation cycle. Modern electro-optic materials can reach operation frequencies of hundreds of gigahertz or even terahertz, far exceeding the gigahertz range of electronic processors [5].

Although photonic computing offers great potential, several **challenges** hinder its practical deployment. First, it relies on high-speed analog optical signals and must be tightly coupled with electronic control, buffering, and digitization due to the absence of mature optical memory. Second, it requires precise synchronization among high-speed Digital-to-Analog Converters (DACs), Analog-to-Digital Converters (ADCs), and photonic devices; even small timing offsets may lead to computation errors and performance degradation [13]. Third, current photonic accelerators lack reconfigurable system-level architectures. Therefore, existing photonic accelerators cannot efficiently support general matrix multiplication (GEMM) or scale beyond small proof-of-concept kernels.

These limitations reveal a clear research gap: **the lack of a programmable and reconfigurable photonic–electronic computing architecture that supports end-to-end GEMM**. Our work bridges this gap by designing and implementing **LightMat**, an efficient hybrid photonic-electronic system for matrix multiplication that integrates precise data stream synchronization and tile-based matrix multiplication. The contributions are summarized as follows:

- We develop **LightMat**, a hybrid photonic–electronic computing system that integrates photonic computing units with an FPGA-based electronic subsystem for end-to-end tile-based matrix multiplication with unsigned 8-bit operands.
- We propose a novel data stream synchronization mechanism that provides precise and reliable data stream alignment while substantially reducing synchronization overhead and improving computational throughput.
- We design a tile-based photonic matrix multiplication scheme to support matrices of arbitrary size.
- We validate synchronisation and numerical accuracy on the single-core prototype and evaluate a 32-group, 1536-lane configuration in simulation.

The rest of the paper is organized as follows: Section 2 gives the background and discusses the related work. Section 3 presents the system design of Light-

Mat. Section 4 describes the system implementation, and Section 5 evaluates the performance of our LightMat system for the evaluated matrix multiplication regime. Section 6 concludes this work and discusses future research.

2 Background and Related Work

2.1 Fundamentals of Photonic Computing

The key principle of photonic computing is to encode digital data onto light using amplitude, phase, or both, and perform computation directly in the optical domain. Mach–Zehnder Modulator (MZM) [6] is a core photonic device for encoding digital signals onto optical carriers. Fig. 1 illustrates continuous amplitude modulation using an MZM. The carrier light is split into two waveguide arms, and the digital data is converted to a Radio Frequency (RF) analog signal, which drives the phase shifters in the two arms and changes their relative phase via the electro-optic effect. When the two paths recombine, their constructive or destructive interference produces the modulated output intensity. The output intensity therefore follows $I_{\text{out}} \propto \cos^2\left(\frac{\Delta\theta}{2}\right)$, where $\Delta\theta = \theta_U - \theta_L$ is the phase difference induced by the differential drive voltage applied to the two arms. Different input voltages (representing different digital data) induce different phase shifts $\Delta\theta$, thereby producing different output intensities.

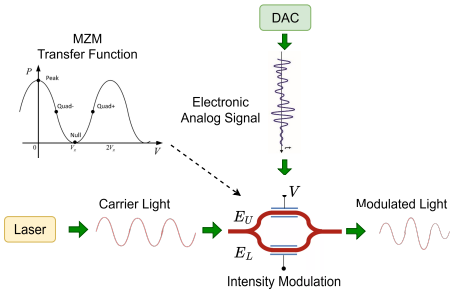

Fig. 1: MZM Intensity modulation.

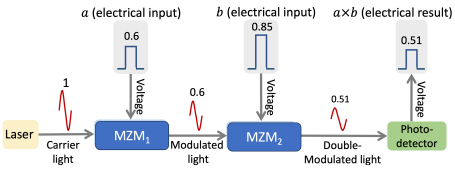

Fig. 2: Illustrations of photonic multiplication.

Photonic Multiplication: As illustrated in Fig. 2, photonic multiplication can be implemented using two cascaded MZMs. The operand a is converted to an RF signal to drive MZM_1, which modulates the optical carrier signal by adjusting its intensity I_{in} as $I_1 = I_{\text{in}} \cdot f(V_1)$ where $f(V_1) = \cos^2\left(\frac{\pi V_1}{2V_\pi}\right)$ is the intensity transfer function of MZM_1, V_1 is the voltage of the RF signal generated to represent a, and V_π is the half-wave voltage required to induce a π phase shift between the two arms. The output of MZM_1 then serves as the input carrier for MZM_2, in which the intensity is further adjusted based on the RF signal representing b as:$I_2 = I_1 \cdot f(V_2) = I_{\text{in}} \cdot f(V_1) \cdot f(V_2)$. When both MZMs operate

in linear regions, $f(V_1) \propto V_1$ and $f(V_2) \propto V_2$, leading to $I_2 \propto V_1 \times V_2$. Therefore, the final output intensity is proportional to $a \times b$.

2.2 Related Work

Photonic Matrix Multiplication: Most photonic matrix-multiplication approaches target multiply-accumulate (MAC) and matrix–vector multiplication (MVM) operations [1]. MZI meshes implement linear transformations through matrix factorization, such as singular value decomposition, offering high generality but limited scalability due to accumulated insertion loss, noise, and errors. WDM-based approaches encode weights with wavelength-selective components, including microring resonators and weight banks, enabling highly parallel and energy-efficient broadcast-and-weight operations [10]. However, thermal sensitivity, wavelength drift, and inter-channel crosstalk complicate weight stabilization and limit numerical precision at scale. Spatial-domain approaches, such as multi-plane light conversion and diffractive optics, provide large-scale linear transformations with high spatial parallelism [1], but generally offer limited reconfigurability and slow weight updates.
Hybrid Photonic-Electronic Architecture: Although photonic computing has been studied for decades, practical systems typically use photonic–electronic hybrid architectures [2, 8], as electronic components remain more efficient for logic control, nonlinear activation, memory access, and synchronization. Accordingly, photonic modules usually perform high-throughput linear operations, such as matrix multiplication, while electronic components handle control flow and nonlinear processing [14, 12]. Lightning is a reconfigurable photonic–electronic SmartNIC that decouples the compute control and data planes using a count-action abstraction [14]. ADEPT performs GEMM operations photonically and non-GEMM operations on a vectorized digital ASIC, but its performance depends on balanced workload partitioning and low electronic control and memory-access latency [3]. Mirage leverages photonic MAC units and residue number systems to achieve high computational precision, but its multi-modulus datapaths and frequent digital conversions incur substantial area and energy overhead [4]. More recent designs, including ROCKET [12] and BITLUME [9], reduce photonic component counts and improve data reuse, yet still require orchestrated electronic pipelines to sustain photonic throughput. Despite these advances, existing systems focus on deep-learning inference or training and do not provide a general system-level framework for accelerating matrix multiplication.

3 LightMat System Design

As shown in Fig. 3, LightMat comprises two subsystems. The *digital subsystem* runs on a Zynq UltraScale+ RFSoC ZCU111, integrating FPGA programmable logic (PL), a processing system (PS), DDR4 memory, and high-performance RF data converters. The *photonic subsystem* performs high-speed optical multiplications. The simplified architecture includes one photonic core with a laser, two

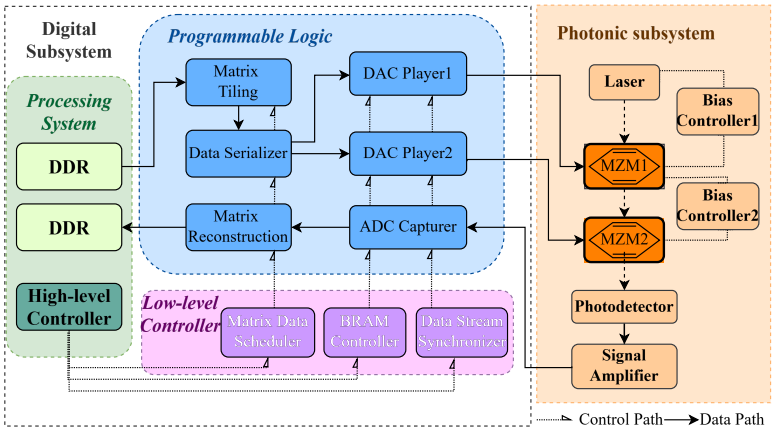

Fig. 3: Simplified schematic of LightMat with one photonic core.

MZMs, and a photodetector, but can scale to multiple cores by adding more MZM pairs with the corresponding *DAC Player* and *ADC Capture* modules.

For matrix multiplication, the tiling module reads matrix blocks from DDR and partitions them into fixed-size tiles for each optical computation cycle. The serializer converts each tile into two synchronized streams for the dual DAC channels, and the *DAC Player* modules continuously stream to the DACs. The *ADC Capture* module digitizes the optical outputs, after which the reconstruction module assembles the partial results and writes the final result to DDR.

3.1 Data Stream Synchronization Mechanism

Data stream synchronization is crucial in hybrid photonic–electronic computing systems because photonic devices operate at much higher frequencies than electronic components. As illustrated in Fig. 4, the electrical voltage representing operand b needs to be applied at the same time as the optical signal representing operand a (*Pulse1*) arrives at MZM2. This temporal alignment ensures that b is properly modulated onto *Pulse1*, enabling the computation of $a \times b$ with the result carried by *Pulse2*. Any misalignment in voltages applied to the two MZMs may lead to incorrect results.

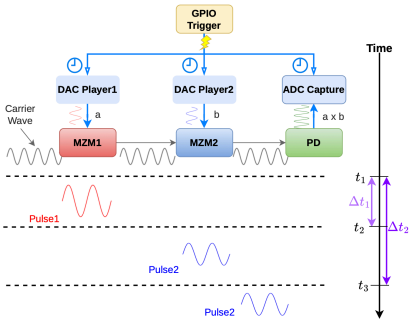

Fig. 4: data stream synchronization and alignment in LightMat

To synchronize DAC data streams, Lightning uses a count-action abstraction [14]. Data transfer begins only when all DACs are ready. However, this

synchronizes the DACs without accounting for propagation delays from MZM1 to MZM2 and the photodetector (PD). As shown in Fig. 4, *Pulse1* must therefore be sufficiently long to ensure that operand b overlaps a valid portion of the pulse, reducing throughput. These delays also require a dedicated preamble to detect valid ADC output, adding further overhead.

We propose a low-overhead data stream alignment mechanism leveraging explicit and proactive hardware-level synchronization. Specifically, we design two modules: *DAC Player* and *ADC Capture*, which are driven by a GPIO (General-Purpose Input/Output) module for simultaneous start and stop of all DACs and ADCs. Each *DAC Player* and *ADC Capture* contains a Block RAM (BRAM) that buffers the data stream to be sent to the MZMs or the data stream received by the photodetector (PD). These two modules share the same clock source with the DACs and ADCs to eliminate clock drift. As illustrated in Fig. 4, to execute a batch of multiplications, the operands are first loaded into the BRAMs of the two *DAC Players*. A single trigger signal is then generated by the GPIO module and broadcast to start the *DAC Players* and the *ADC Capture* simultaneously.

To compensate for the propagation delays from MZM1 to MZM2 and to the PD, timing offsets are introduced to ensure precise alignment. As illustrated in Fig. 4, suppose that *DAC Player1* starts streaming a at time t_1, and *Pulse1* arrives at MZM2 at time t_2, and *Pulse2* arrives at PD at t_3. Then $\Delta t_1 = t_2 - t_1$ is the propagation delay from MZM1 to MZM2, and $\Delta t_2 = t_3 - t_1$ is the propagation delay from MZM1 to the PD. Let $N(\Delta t_1)$ and $N(\Delta t_2)$ be the number of data samples generated by the DACs in Δt_1 and Δt_2, respectively. Both values are measured during system calibration. To ensure that the voltages corresponding to both operands are applied to the same optical pulse, the streaming of operand b is pre-offset by $N(\Delta t_1)$ samples. Similarly, the valid ADC output is captured with an offset of $N(\Delta t_2)$ samples per trigger. Experimental validation on our testbed shows that Δt_1 and Δt_2 remain stable (see details in Section 5.1).

3.2 Tile-based Matrix Multiplication

Each *DAC Player* module includes a BRAM buffer to stream operands to the DACs at high speed. On the ZCU111 board, each BRAM is limited to 32KB, which in turn limits the number of multiplication operations that can be executed per trigger. To support matrix multiplication of arbitrary size, LightMat uses a tile-based matrix multiplication scheme in which a large matrix is partitioned into fixed-size $T \times T$ tiles. The tile size T is determined by ensuring that the operands for all multiplications of a tile pair can be loaded into the BRAMs of the two *DAC Players* so that the multiplication of a tile pair can be executed with a single trigger. The tiled photonic matrix multiplication is illustrated in Fig. 5, and the pseudocode is given in Algorithm 1.

Matrix Tiling: Given $A \in \mathbb{R}^{M \times K}$, $B \in \mathbb{R}^{K \times N}$ with output $C = AB \in \mathbb{R}^{M \times N}$, and a tile size T, A and B are padded to the nearest multiple of T as follows:

$$\hat{M} = \left\lceil \frac{M}{T} \right\rceil T, \hat{K} = \left\lceil \frac{K}{T} \right\rceil T, \hat{N} = \left\lceil \frac{N}{T} \right\rceil T, \tag{1}$$

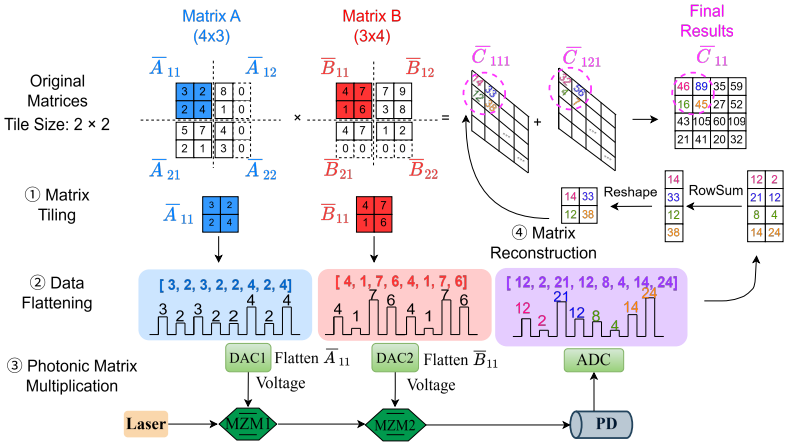

Fig. 5: Illustration of the tile-based photonic matrix multiplication.

We construct $\bar{A} \in \mathbb{R}^{\hat{M}\times\hat{K}}$ and $\bar{B} \in \mathbb{R}^{\hat{K}\times\hat{N}}$ by padding A and B with zeros along the bottom and right boundaries:

$$\bar{A}_{i,j} = \begin{cases} A_{i,j}, & 1 \le i \le M,\ 1 \le j \le K, \\ 0, & \text{otherwise}, \end{cases} \tag{2}$$

$$\bar{B}_{i,j} = \begin{cases} B_{i,j}, & 1 \le i \le K,\ 1 \le j \le N, \\ 0, & \text{otherwise}. \end{cases} \tag{3}$$

This zero padding preserves correctness because any product involving a padded entry contributes zero to the accumulation. We tile $\bar{A}$, $\bar{B}$, and the padded output $\bar{C} = \bar{A}\bar{B} \in \mathbb{R}^{\hat{M}\times\hat{N}}$ into $T \times T$ blocks:$\bar{A} = [\bar{A}_{p,q}], \bar{B} = [\bar{B}_{q,r}], \bar{C} = [\bar{C}_{p,r}]$, where $1 \le p \le \hat{M}/T$, $1 \le q \le \hat{K}/T$, $1 \le r \le \hat{N}/T$, and $\bar{A}_{p,q}, \bar{B}_{q,r}, \bar{C}_{p,r} \in \mathbb{R}^{T\times T}$.

Data Flattening and Photonic Tile Multiplication: each tile pair is transformed into one-dimensional sequences to match the temporal input format of the two DACs. For $\bar{A}_{ik} \times \bar{B}_{kj}$, $\bar{A}_{ik}$ is flattened in a row-major order, with each row repeated T times before flattening the next row. In contrast, $\bar{B}_{kj}$ is flattened in a column-major manner: the entire matrix is first unfolded column by column, and the resulting vector is then repeated T times. As illustrated in Fig. 5, $\bar{A}_{11}$ is flattened to $V_1 = [3, 2, 3, 2, 2, 4, 2, 4]$, and $\bar{B}_{11}$ is flattened to $V_2 = [4, 1, 7, 6, 4, 1, 7, 6]$. This complementary row-major and column-major flattening ensures that each pair of samples streamed to the dual DAC channels corresponds to an element-wise product between one row of $\bar{A}_{11}$ and one column of $\bar{B}_{11}$, forming the partial results of the tile-based matrix multiplication.

Matrix Reconstruction: the reconstruction process for tile-based matrix multiplication can be represented as $\bar{C}_{p,r} = \sum_{q=1}^{\hat{K}/T} \bar{A}_{p,q}\bar{B}_{q,r}$, where $1 \le p \le \hat{M}/T$ and $1 \le r \le \hat{N}/T$. Algorithm 2 gives the pseudocode for reconstructing the

Input: $A \in \mathbb{R}^{M\times K}$, $B \in \mathbb{R}^{K\times N}$
T: tile size.
Output: C: result matrix
1 **Step 1: Matrix Tiling:**
// Pad the input matrices
2 $\bar{A} = \mathrm{Pad}(A), \bar{B} = \mathrm{Pad}(B)$.
// Decompose $\bar{A}$, $\bar{B}$ into $T\times T$ tiles
3 $\bar{A} = [\bar{A}_{i,k}], \quad \bar{B} = [\bar{B}_{k,j}], \quad 1 \le i,j,k \le \hat{N}/T.$
4 **Step 2: Data Flattening:**
5 **foreach** *tile pair* $(\bar{A}_{ik}, \bar{B}_{kj})$ **do**
6 $V_1 = \mathrm{Flatten}_r(\bar{A}_{i,k}, T)$;
7 $V_2 = \mathrm{Flatten}_c(\bar{B}_{k,j}, T)$;
8 Append (V_1, V_2) to buffer $\mathcal{B}$
9 **Step 3: Photonic Multiplication:**
10 **foreach** *voltage pair* (V_1, V_2) *in* $\mathcal{B}$ **do**
11 **for** $t \leftarrow 1$ ***to*** *length*(V_1) **do**
12 $res \leftarrow \mathrm{PhotonicMultiply}(V_1[t], V_2[t])$
13 Append res to result buffer $\mathcal{R}$
14 **Step 4: Matrix Reconstruction:**
// Aggregate tile multiplications
15 $C = \mathrm{MatrixReconstruct}(\mathcal{R}, T, M, K, N)$
16 **return** C

Algorithm 1: MatrixMultiply(A, B, T)

Input: $\mathcal{R}$: pair-wise multiplication results;
T: tile size; $[M, K], [K, N]$: matrix dimensions.
Output: C: reconstructed matrix
// Initialization
1 $r \leftarrow \lceil M/T \rceil, c \leftarrow \lceil N/T \rceil, d \leftarrow \lceil K/T \rceil$
2 $M[R_t][C_t] = 0$;
3 $idx \leftarrow 0$;
4 **Step 1: Tile-wise reconstruction**
5 **for** $i \leftarrow 1$ ***to*** r **do**
6 **for** $j \leftarrow 1$ ***to*** c **do**
7 **for** $q \leftarrow 1$ ***to*** d **do**
8 $flat \leftarrow \mathcal{R}[idx : idx + T^3]$;
9 $idx \leftarrow idx + T^3$;
10 $s \leftarrow \mathrm{Reshape}(flat, (-1, T))$;
11 $s \leftarrow \mathrm{RowSum}(s)$;
12 $R_{tile} \leftarrow \mathrm{Reshape}(s, (T, T))$;
13 $M[i][j] \leftarrow M[i][j] + R_{tile}$;
14 **Step 2: Combine sub-matrices**
15 $\bar{C} \leftarrow \mathrm{BlockConcat}(M)$;
16 **Step 3: Crop valid region**
17 $C \leftarrow \bar{C}[1{:}M,\ 1{:}N]$;
18 **return** C;

Algorithm 2: MatrixReconstruct(R,T,M,K,N)

result matrix C. The ADC output for each trigger is a long one-dimensional sequence that contains the results of all individual multiplications for multiplying a pair of tiles. Let $\mathcal{R}$ be a one-dimensional array that contains the multiplication results of all tile pairs. Since each tile has size T^2 and each element is repeated T times during flattening, the multiplication of a single tile pair contains T^3 multiplication results (line 8). To obtain the tile-level results, these T^3 elements are first shaped into a $T^2 \times T$ matrix (line 10), with each row summing up to a vector of T^2 elements. The vector is then reshaped to a $T \times T$ matrix block. Each output tile is obtained by accumulating the corresponding tile products along the reduction dimension q (lines 7-13). This process can be illustrated using the example in Fig. 5. The ADC output for $\bar{A}_{11} \times \bar{B}_{11}$ [12, 2, 21, 12, 8, 4, 14, 24] is first reshaped into a 4×2 matrix and then summed along each row. The resulting 4-element vector is reshaped to a 2×2 matrix $\bar{C}_{111}$. Similarly, $\bar{A}_{12} \times \bar{B}_{21}$ produces $\bar{C}_{121}$. The final output tile C_{11} is obtained by accumulation: $\bar{C}_{11} = \bar{A}_{11}\bar{B}_{11} + \bar{A}_{12}\bar{B}_{21} = \bar{C}_{111} + \bar{C}_{121}$. After all tiles are reconstructed, they are concatenated to form the padded result $\bar{C}$ (line 15). The final matrix C is obtained by cropping the upper-left $M \times N$ region of $\bar{C}$ (line 17).

Buffer and reconstruction constraints. A $T \times T$ tile pair requires T^3 scalar products. If each product is explicitly represented by a 50-sample waveform and each DAC sample occupies 16 bits, each DAC buffer must hold $100T^3$ bytes. For a 32 KiB buffer, this bounds the pre-expanded tile size to $T \le 6$; $T = 16$ would exceed capacity unless waveform repetition or additional buffering is performed in hardware. After each tile-pair capture, samples are converted

into scalar products, summed along the local reduction dimension, and the reconstructed tile is accumulated across q on the processing system.

4 Implementation

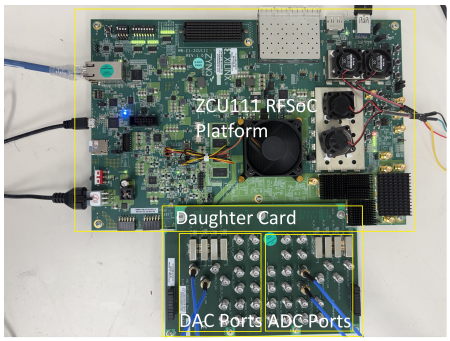

Fig. 6: Digital subsystem implemented on the AMD Zynq UltraScale+ RFSoC ZCU111 evaluation board.

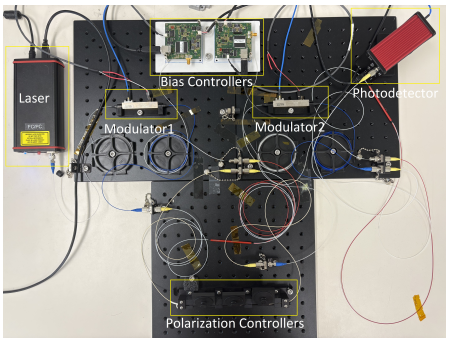

Fig. 7: Photonic subsystem consisting of laser, MZMs, bias controller, polarization controller, and photodetector.

Digital Components: The digital subsystem is implemented on the Xilinx Zynq UltraScale+ RFSoC ZU28DR of the ZCU111 board. The DACs operate in single-ended mode with an output voltage of approximately 1 Vpp. Since the optical modulator requires $V_\pi = 5$ Vpp, the output is amplified using a Pasternack PE15A63018 RF amplifier. On the receiving side, the photodetector output is a few hundred millivolts, which is below the ADC input range. An LMH5401 amplifier is used to both amplify the signal and add the required $V_{\text{cm}} = 1.25$ V common-mode voltage for DC-coupled operation.

Photonic Components: The photonic subsystem uses commercial optical components. A tunable continuous-wave laser generates a carrier at $\lambda = 1550$ nm, while two Thorlabs 10-GHz intensity modulators encode the electrical signals onto the optical carrier. A Thorlabs polarization controller between the modulators maintains the polarization state, and two MBC-SUPER bias controllers lock them at the null points of their intensity transfer functions. The modulated light is detected by a Thorlabs RXM15EF photodetector with a 15-GHz bandwidth.

The *LightMat* prototype is implemented in Vivado 2024.1. Its programmable logic (PL) implements matrix tiling, data serialization, *DAC Player*, *ADC Capture*, and matrix reconstruction, while PYNQ provides software control for matrix input and result retrieval. The *DAC Player* and *ADC Capture* modules combine AXI BRAM Controller and Block Memory Generator IPs with custom RTL. An AXI GPIO distributes synchronization triggers to all modules. AXI control operates at 100 MHz; DAC streaming and ADC capture use 256 MHz and

512 MHz clocks, respectively. AXI4-Stream is used to support high-throughput data transfer. Both DACs and ADCs operate at 4.096 GS/s.

5 Performance Evaluation

5.1 Synchronization Robustness

To measure the delay offsets in our synchronization scheme and evaluate its robustness, we conduct experiments using a specially designed data sequence shown in Fig. 8 (a). In this special sequence, the signal value steps from 0 to 9000 at sample 300, and from 9000 to 12000 at sample 1300. Both DACs are configured to operate in loopback mode to continually convert the last data sample until the next trigger.

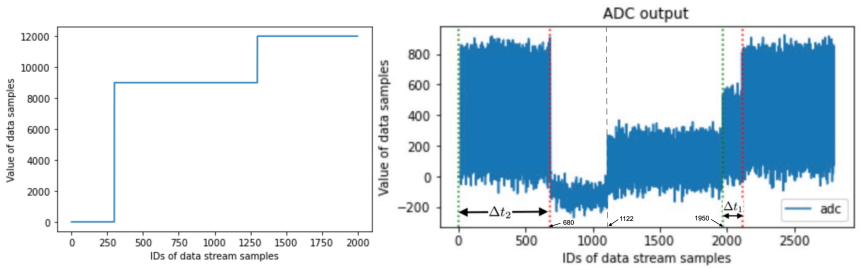

Fig. 8: (a) Data stream of DAC input. (b) Data stream of ADC output.

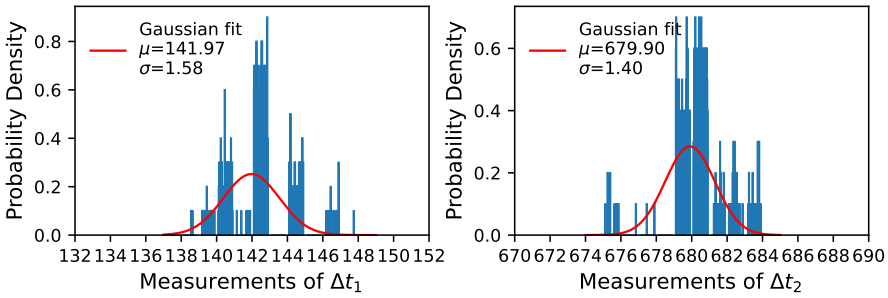

Fig. 9: Distribution of $N(\Delta t_1)$ and $N(\Delta t_2)$

Fig. 8 (b) shows the data captured by the ADC. The interval from sample 680 to 1122 corresponds to the period during which at least one DAC input is zero. Its length exceeds 300 samples due to the propagation delay from MZM1 to MZM2. The period from sample 0 to 680 corresponds to the delay from MZM1 to PD,

i.e., $N(\Delta t_2)$. During this period, the ADC output remains high because it is still capturing residual results from the previous trigger. The plateau from sample 1122 to 1950 corresponds to the interval in which MZM2 is modulating 9000 on the pulse that already carries 9000 modulated by MZM1, i.e., 9000×9000. The small plateau after sample 1950 corresponds to the interval beginning when both DACs start modulating 12000 and ending just before the optical signal carrying 12000 modulated by MZM1 reaches MZM2, and the effective multiplication is 9000×12000. Obviously, this interval represents the delay from MZM1 to MZM2 (Δt_1). To evaluate the stability of Δt_1 and Δt_2, we conducted 500 measurements collected over 10 experimental runs. In each run, the system was first calibrated, and the streaming of the sequence shown in Fig. 8 was then triggered 50 times. Fig. 9 shows the distribution of $N(\Delta t_1)$ and $N(\Delta t_2)$ over 500 measurements. The mean values of $N(\Delta t_1)$ and $N(\Delta t_2)$ are 142 and 680 samples, respectively, with standard deviations below 2 samples, demonstrating high stability across 500 measurements. These offsets mainly depend on cable lengths, with slight variations caused by phase misalignment between the two MZMs.

5.2 Evaluation of Accuracy on Matrix Multiplication

We use the **Mean Relative Error (MRE)** defined as follows to evaluate the accuracy of matrix multiplication.

$$MRE = \frac{1}{N} \sum_{i=1}^{N} \frac{|y^{(i)}_{\text{measured}} - y^{(i)}_{\text{ground_truth}}|}{|y^{(i)}_{\text{ground_truth}}|}, \tag{4}$$

where N is the number of multiplications in a matrix multiplication. The size of the matrix is varied from 16×16 to 1024×1024. Matrix elements are drawn uniformly from positive integers in $[1, 255]$. Signed and floating-point arithmetic may be extended using sign decomposition and photonic mantissa multiplication [9, 4]; these operations remain future evaluation targets.

Fig. 10 shows the MRE and its standard deviation over different matrix sizes. The MRE remains stable across different matrix dimensions, indicating that the photonic computing pipeline maintains consistent accuracy as the scale grows. The slightly higher MRE observed for the 16×16 case is expected, as the limited number of elements provides less statistical averaging, making the MRE more sensitive to element-level relative errors. From 16×16 to 1024×1024, the MRE remains around 0.028 with a standard deviation of ∼0.001, showing that tiling and digital accumulation do not increase the error with matrix size.

An MRE of approximately 2.8% corresponds to about 5 bits of relative precision, targeting error-tolerant workloads such as quantized DNN inference, for which we set the application requirement as at most one percentage-point top-1 accuracy loss relative to an equally quantized digital reference.

5.3 Performance Comparison with Baselines

Sections 5.1–5.2 report measurements on the single-core prototype; this section evaluates a 32-group configuration in simulation. The baselines comprise

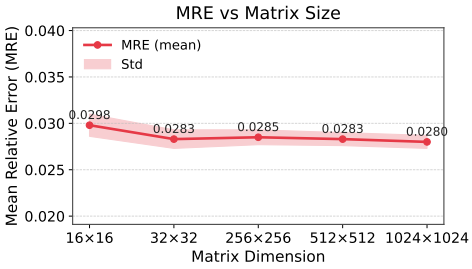

Fig. 10: Measured, single-core: Mean Relative Error

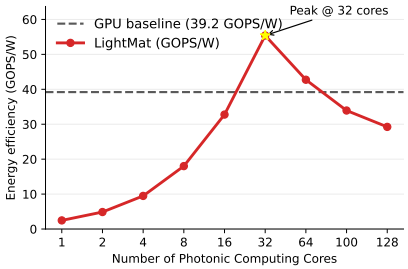

Fig. 11: Simulated: Energy Efficiency Scaling

an NVIDIA A100 40-GB PCIe GPU using cuBLAS, an HLS GEMM kernel on ZCU111, and Lightning [14].

Simulation configuration. We simulate 32 core groups, each with 48 parallel scalar-product lanes (1536 lanes in total), at an effective lane rate of 97 Gsamples/s. This scaled configuration assumes integrated thin-film lithium niobate modulators with electro-optic bandwidth above 100 GHz [11] and corresponding high-speed electronic interfaces. The prototype supplies the measured sample lengths and alignment offset: each LightMat product occupies $s = 50$ samples, whereas the Lightning-style comparison uses $s = 192$ to accommodate the 142-sample offset. The throughput is $Q = 2nf_s u/s$; with simulated utilisation $u \approx 0.65$, LightMat achieves approximately 3.9 TOPS. The 142-sample offset and 50-sample product window establish the simulator's pulse-timing calibration from the single-core measurements.

Transfers and reconstruction. For each output tile (p, r), the tile products are streamed in reduction order q, the T^3 scalar products are reduced to T^2 output values, and the resulting tiles are accumulated before DDR write-back. With 16-bit stored samples and a fully expanded 50-sample pulse for each product, each DAC input buffer requires $100T^3$ bytes; thus a 32-KiB buffer permits $T \leq 6$ without on-the-fly sample repetition. We assume an effective shared off-chip bandwidth $B = 24$ GB/s, and model latency as $L \geq \max(D/B, L_{\text{opt}}) + L_{\text{ctrl}} + L_{\text{rec}}$, where D denotes off-chip traffic, L_{ctrl} covers tile-batch triggering and PS–PL coordination, and L_{rec} covers post-capture reduction and write-back, both serial to the overlapped transfer/optical stage. For a 512×512 GEMM, the minimum 16-bit data traffic is $D_{\min} = 2(MK+KN+MN) \approx 1.6$ MB, requiring 65.5 μs at the assumed bandwidth; the optical compute time at 3.9 TOPS is 68.9 μs. The balance between bandwidth and computation motivates the 32-group scaling point in Fig. 11.

A100 baseline and power sensitivity. The A100 comparison point uses 9800 GOPS on 512×512 GEMM (27.4 μs at two operations per multiply–accumulate). Assigning the PCIe TDP of 250 W gives 39.2 GOPS/W. At fixed throughput, 150 W and 200 W give 65.3 and 49.0 GOPS/W; the LightMat efficiency ratios at 150, 200, and 250 W are 0.8×, 1.1×, and 1.4×, respectively. Fu-

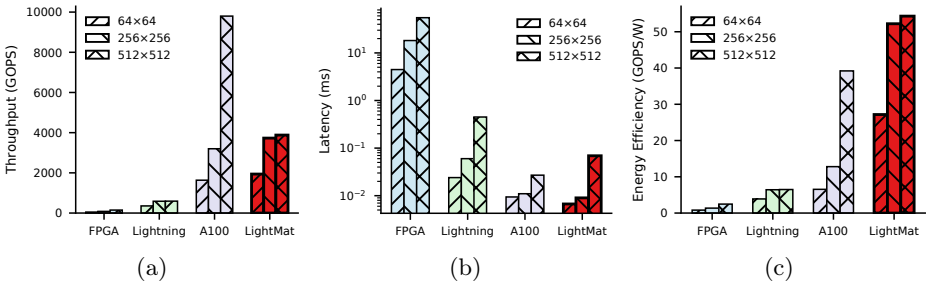

Fig. 12: Simulation comparisons: Performance comparison across FPGA, Lightning, GPU, and LightMat under different matrix sizes.

ture work will benchmark the A100 at matched accuracy, reporting the cuBLAS arithmetic mode, the output MRE, and the NVML-measured workload power.

Power accounting. The single-core prototype's listed component subtotal is approximately 13.3 W: 6.5 W for ZCU111, 6.76 W for two discrete bias controllers, and less than 40 mW for the optical devices; the laser electrical supply and RF amplification contribute additional power. For the integrated 1536-lane simulation, we use $P(n) = P_{\text{shared}} + nP_{\text{inc}}$, where $P_{\text{shared}} = 20$ W comprises 12 W for control, memory and converter infrastructure, 4 W for the laser electrical supply, and 4 W for shared integrated bias control. The incremental budget is $P_{\text{inc}} \approx 33.7$ mW per lane for RF drive, receiver and per-lane control, giving $P(1536) \approx 71.8$ W. Shared integrated bias control replaces the prototype's discrete 3.38-W-per-controller scheme. At fixed throughput, an approximately 40% increase in LightMat power would remove its 1.4× efficiency over the A100.

Simulation findings. LightMat achieves 6.5× higher throughput and 7.0–8.4× higher energy efficiency than Lightning, and 1.4–4.2× higher energy efficiency than the A100 under the assumed baseline power, reaching 54.3 GOPS/W.

6 Conclusions and Future Work

We proposed LightMat, a photonic-electronic system for tile-based unsigned 8-bit matrix multiplication, combining optical modulation with programmable digital control. The streaming architecture is designed to enable sustained photonic computation. The single-core prototype exhibits stable timing offsets over 500 measurements and approximately 2.8% MRE for positive unsigned 8-bit inputs. In simulation, a 32-group, 1536-lane LightMat achieves 6.5× higher throughput and over 7× higher energy efficiency than Lightning, as well as 1.4× the energy efficiency of an A100 at the assumed 250-W operating point for 512×512 GEMM. Future work will implement the integrated multi-lane architecture, measure complete system and workload-dependent GPU power, calibrate end-to-end latency, and extend validation to signed and floating-point operands.

References

1. Cheng, J., Zhou, H., Dong, J.: Photonic matrix computing: from fundamentals to applications. nanomaterials **11**(7), 1683 (2021)
2. Chengpeng Xia, Yawen Chen, H.Z., Wu, J.: Tadia: Photonic stochastic gradient descent for neural network accelerators. ACM Transactions on Embedded Computing Systems **22**(5s), 1–23 (2023)
3. Demirkiran, C., Eris, F., Wang, G., Elmhurst, J., Moore, N., Harris, N.C., Basumallik, A., Reddi, V.J., Joshi, A., Bunandar, D.: An electro-photonic system for accelerating deep neural networks. ACM journal on emerging technologies in computing systems **19**(4), 1–31 (2023)
4. Demirkiran, C., Yang, G., Bunandar, D., Joshi, A.: Mirage: An rns-based photonic accelerator for dnn training. In: 2024 ACM/IEEE 51st Annual International Symposium on Computer Architecture (ISCA). pp. 73–87. IEEE (2024)
5. Lampert, Y., Shams-Ansari, A., Gaier, A., Tomasino, A., Cao, X., Magalhaes, L., Rajabali, S., Lončar, M., Benea-Chelmus, I.C.: Photonics-integrated terahertz transmission lines. Nature Communications **16**(1), 7004 (2025)
6. Saleh, B.E., Teich, M.C.: Fundamentals of photonics, 2 volume set. john Wiley & sons (2019)
7. Sîrbu, A., Babaoglu, O.: Power consumption modeling and prediction in a hybrid cpu-gpu-mic supercomputer. In: European Conference on Parallel Processing. pp. 117–130. Springer (2016)
8. Xia, C., Zhang, H., Chen, Y., Barnard, A.S.: Siphon: Silicon photonic computing based chiplet accelerator for deep learning. IEEE journal on emerging and selected topics in circuits and systems (2025)
9. Xia, C., Zhang, H., Zhang, H., Chen, Y., Barnard, A.: Bitlume: Precision-flexible photonic computing for ultra-fast and energy-efficient dnn acceleration. In: Proceedings of International Conference on Computer-Aided Design (ICCAD) (2025)
10. Xu, X., Tan, M., Corcoran, B., Wu, J., Boes, A., Nguyen, T.G., Chu, S.T., Little, B.E., Hicks, D.G., Morandotti, R., et al.: 11 tops photonic convolutional accelerator for optical neural networks. Nature **589**(7840), 44–51 (2021)
11. Yamaguchi, Y., Dat, P.T., Yamamoto, N., Akahane, K., Kanno, A., Kawanishi, T.: Thin-film lithium niobate modulator for a flat frequency-response over 110 ghz bandwidth with integrated electro-optic frequency-domain equalizer. In: Optical Fiber Communication Conference (OFC). p. M3K.4. Optica Publishing Group (2024). https://doi.org/10.1364/OFC.2024.M3K.4
12. Zhang, H., Zhang, H., Xia, C., Huang, Z., Chen, Y., Barnard, A.: Rocket: An rns-based photonic accelerator for high-precision and energy-efficient dnn training. In: Proceedings of the 39th ACM International Conference on Supercomputing. pp. 1020–1033 (2025)
13. Zhang, X., Zhang, X., Chen, Y., Jin, W., Zhou, Z., Liu, C., Lao, C., Huang, J., Dong, J., Ma, W., et al.: Microcomb-synchronized optoelectronics. Nature Electronics pp. 1–9 (2025)
14. Zhong, Z., Yang, M., Lang, J., Williams, C., Kronman, L., Sludds, A., Esfahanizadeh, H., Englund, D., Ghobadi, M.: Lightning: A reconfigurable photonic-electronic smartnic for fast and energy-efficient inference. In: Proceedings of the ACM SIGCOMM Conference. pp. 452–472 (2023)