

University of Kentucky

UKnowledge

Theses and Dissertations--Electrical and
Computer Engineering

Electrical and Computer Engineering


2024

Cross-Layer Design of Highly Scalable and Energy-Efficient AI Accelerator Systems Using Photonic Integrated Circuits

Sairam Sri Vatsavai

University of Kentucky, ssr226@uky.edu

Author ORCID Identifier:

 <https://orcid.org/0000-0003-1847-3976>

Digital Object Identifier: <https://doi.org/10.13023/etd.2024.143>

[Right click to open a feedback form in a new tab to let us know how this document benefits you.](#)

Recommended Citation

Sri Vatsavai, Sairam, "Cross-Layer Design of Highly Scalable and Energy-Efficient AI Accelerator Systems Using Photonic Integrated Circuits" (2024). *Theses and Dissertations--Electrical and Computer Engineering*. 201.

https://uknowledge.uky.edu/ece_etds/201

This Doctoral Dissertation is brought to you for free and open access by the Electrical and Computer Engineering at UKnowledge. It has been accepted for inclusion in Theses and Dissertations--Electrical and Computer Engineering by an authorized administrator of UKnowledge. For more information, please contact UKnowledge@lsv.uky.edu.

STUDENT AGREEMENT:

I represent that my thesis or dissertation and abstract are my original work. Proper attribution has been given to all outside sources. I understand that I am solely responsible for obtaining any needed copyright permissions. I have obtained needed written permission statement(s) from the owner(s) of each third-party copyrighted matter to be included in my work, allowing electronic distribution (if such use is not permitted by the fair use doctrine) which will be submitted to UKnowledge as Additional File.

I hereby grant to The University of Kentucky and its agents the irrevocable, non-exclusive, and royalty-free license to archive and make accessible my work in whole or in part in all forms of media, now or hereafter known. I agree that the document mentioned above may be made available immediately for worldwide access unless an embargo applies.

I retain all other ownership rights to the copyright of my work. I also retain the right to use in future works (such as articles or books) all or part of my work. I understand that I am free to register the copyright to my work.

REVIEW, APPROVAL AND ACCEPTANCE

The document mentioned above has been reviewed and accepted by the student's advisor, on behalf of the advisory committee, and by the Director of Graduate Studies (DGS), on behalf of the program; we verify that this is the final, approved version of the student's thesis including all changes required by the advisory committee. The undersigned agree to abide by the statements above.

Sairam Sri Vatsavai, Student

Ishan G Thakkar, Major Professor

Daniel Lau, Director of Graduate Studies

Cross-Layer Design of Highly Scalable and Energy-Efficient AI Accelerator Systems
Using Photonic Integrated Circuits

DISSERTATION

A dissertation submitted in partial
fulfillment of the requirements for
the degree of Doctor of Philosophy
in the College of Engineering at the
University of Kentucky

By
Sairam Sri Vatsavai
Lexington, Kentucky

Director: Dr. Ishan G Thakkar,
Assistant Professor of Electrical and Computer Engineering
Lexington, Kentucky
2024

Copyright© Sairam Sri Vatsavai 2024

ABSTRACT OF DISSERTATION

Cross-Layer Design of Highly Scalable and Energy-Efficient AI Accelerator Systems Using Photonic Integrated Circuits

Artificial Intelligence (AI) has experienced remarkable success in recent years, solving complex computational problems across various domains, including computer vision, natural language processing, and pattern recognition. Much of this success can be attributed to the advancements in deep learning algorithms and models, particularly Artificial Neural Networks (ANNs). In recent times, deep ANNs have achieved unprecedented levels of accuracy, surpassing human capabilities in some cases. However, these deep ANN models come at a significant computational cost, with billions to trillions of parameters. Recent trends indicate that the number of parameters per ANN model will continue to grow exponentially in the foreseeable future. To meet the escalating computational demands of ANN models, the hardware accelerators used for processing ANNs must offer lower latency and higher energy efficiency. Unfortunately, traditional electronic implementations of ANN hardware accelerators, including CPUs, Graphics Processing Units (GPUs), Application-Specific Integrated Circuits (ASICs), and Field Programmable Gate Arrays (FPGAs), have fallen short of meeting the latency and energy efficiency requirements for processing deep ANN models. Furthermore, the interconnection network subsystems in these electronic accelerator systems, designed to facilitate large-scale data transfers between processing cores and memory/control units within the accelerator systems, have become bottlenecks that hinder the throughput, latency, and energy efficiency of deep ANN model processing.

Fortunately, Photonic Integrated Circuits (PICs)-based accelerator systems, featuring photonic network subsystems are promising alternatives to conventional electronic accelerators. PIC-based accelerator systems operate in the optical domain, delivering processing at the speed of light with ultra-low latency, minimal dynamic energy consumption, and high throughput. These advantages stem from the wavelength division multiplexing capabilities and the absence of distance-dependent impedance in PICs. Furthermore, these characteristics enable the implementation of high-performance photonic network subsystems within PIC-based accelerator systems. Additionally, PIC-based accelerator systems offer inherent optical nonlinearities.

Despite these numerous advantages over electronic accelerators, PIC-based systems still encounter several challenges due to limited optical power budget, susceptibility to crosstalk and other sources of noise caused by the analog operation, high area consumption, and restricted functional flexibility of PICs. These challenges manifest in various ways. (i) The existence of a significant trade-off between the achievable processing core size and the supported bit precision that impedes the scalability of processing cores. (ii) The limited reconfigurability, in terms of supported computing size and precision, makes them less adaptable to modern ANN models with diverse computational and precision demands. (iii) The reliance on electronic adder networks for accumulation diminishes the latency and energy consumption benefits of PIC-based accelerator systems due to frequent analog-to-digital conversions and memory accesses involved in accumulations.

My research has contributed several solutions that overcome a multitude of these challenges and improve the throughput, energy efficiency, and flexibility of PIC-based AI accelerator systems. I identified and analyzed factors that affect the scalability and reconfigurability of PIC-based AI accelerator systems. I proposed several novel PIC-based accelerator architectures with enhancements at the circuit level, architecture level, and system level to improve scalability, reconfigurability, and functional flexibility. At the circuit level, these enhancements serve to decrease optical signal losses, reduce control complexity, enable adaptability for various ANN processing tasks, and lower power and area consumption. The architecture-level improvements mitigate crosstalk noise, facilitate functional reconfigurability, enable in-situ and flexible spatio-temporal accumulation, and provide flexible support for different dataflows. The system-level enhancements involve the integration of stochastic computing with PIC-based accelerators to break the inherent trade-off between scalability and supported bit precision. Additionally, applying stochastic computing enhances the flexibility of PIC-based accelerators, allowing them to support mixed-precision ANN models. These cross-layer enhancements collectively contribute to the design of PIC-based AI accelerator systems, resulting in improved throughput, energy efficiency, scalability, and reconfigurability.

KEYWORDS: Photonic Computing, Photonic AI Accelerators, Photonic Interconnects, Reservoir Computing, AI Hardware

Sairam Sri Vatsavai

May 6, 2024

Cross-Layer Design of Highly Scalable and Energy-Efficient AI Accelerator Systems
Using Photonic Integrated Circuits

By
Sairam Sri Vatsavai

Dr. Ishan G Thakkar
Director of Dissertation

Dr. Daniel Lau
Director of Graduate Studies

May 6, 2024

Date

In memory of my taatayya (grandfather), Vijay Gopal Raju, whose wisdom and love continue to inspire me. This thesis is a testament to the values you instilled in me.

ACKNOWLEDGMENTS

I would like to express my deepest gratitude to my advisor, Dr. Ishan Thakkar, for his exceptional support and guidance throughout my PhD journey. His compassionate and discerning nature proved truly invaluable. He possessed a remarkable ability to understand my struggles and offer support while also providing insightful feedback that challenged me to grow. These qualities, along with Dr. Thakkar's unwavering transparency, fostered an exceptional learning environment. His willingness to share not only successes but also challenges in the research process deepened my understanding of the realities of academic research. Beyond academic guidance, I am especially grateful to Dr. Thakkar for the many life lessons imparted throughout my PhD journey. He generously shared valuable insights on perseverance and navigating professional challenges, which will undoubtedly serve me well beyond the confines of this dissertation.

My journey wouldn't have been nearly as enriching without the steadfast support of a dear friend. To Praneeth Karempudi, my heartfelt appreciation. You played a pivotal role in illuminating the path to my doctoral journey, offering unwavering companionship and collaboration every step of the way.

I extend my gratitude to my esteemed committee members, Dr. Todd Hastings, Dr. Luis Giraldo, and Dr. Simone Silvestri, for their invaluable support and guidance.

I also sincerely thank my colleagues in the Unconventional Computing Architectures and Technologies (UCAT) lab—Chao-Hsuan Huang, Oluwaseun Alo, Samrat Patel, Bobby Bose, and David Phippen for their help and support.

I am profoundly grateful to my parents and my sister, Lalitha Sri Vatsavai, for their unwavering encouragement and boundless love, which have been the cornerstone of my journey. Additionally, I extend my heartfelt appreciation to my cherished paw-

sibling, Jack, whose loyal companionship has been a source of constant comfort and joy.

Lastly, I extend heartfelt gratitude to my exceptional friends Sai Praneeth Reddy, Sai Charan, Shiva Prasad, Santhosh Pogaku, Sai Teja, Amit Degada, Bhamiti Sharma, Shravani Prakhya, Deepak Kumar, Srinivasa Rao, Prakash Dhungana, Ankan Bhat-tacharya, Sindhuja, Pragnya, Amandeep Garg, and Simran Goyal for their steadfast support and encouragement throughout my doctoral pursuit. Their unwavering help has been instrumental in navigating the challenges and joys of this journey. There is not enough room in this dissertation to express the depth of their support, but they should know they are a major reason for the completion of this work.

TABLE OF CONTENTS

Acknowledgments	iii
List of Tables	x
List of Figures	xii
Chapter 1 Introduction	1
1.1 Artificial Neural Networks	1
1.1.1 Feed Forward Neural Networks	2
1.1.2 Convolutional Neural Networks	2
1.1.3 Recurrent Neural Networks	4
1.2 Need for ANN Accelerators	4
1.3 Photonic Integrated Circuits Based AI Accelerators	6
1.3.1 Fundamentals: Microring Resonator	9
1.4 Challenges of PIC based AI Accelerators	13
1.4.1 Reservoir Computing Challenges	13
1.4.2 Dot Product Acceleration Challenges	16
1.4.3 Photonic Network Subsystems Challenges	19
1.5 Contributions	20
Chapter 2 Silicon Photonic Microring Based Chip-Scale Accelerator for De- layed Feedback Reservoir Computing	23
2.1 Introduction	23
2.2 Related Work And Motivation	25
2.2.1 DFRC Accelerators from Prior Work	25
2.2.2 Motivation for MR-based Chip-Scale DFRC Accelerators	25
2.3 Background and Fundamentals	26
2.3.1 Delayed Feedback Reservoir Computing (DFRC)	26
2.3.2 Fundamentals: Microring Resonators (MRs)	28
2.4 Proposed MR-based Chip-Scale DFRC Accelerator	29
2.4.1 Overview	29
2.4.2 Modelling the Nonlinear Response of the Reservoir MR	30
2.5 Evaluation	31
2.5.1 Evaluation Setup	31
2.5.2 Error Metrics	31
2.5.3 Prediction Error Evaluation	32
2.5.4 Training Time	34
2.5.5 Discussion on Power Consumption	35
2.6 Summary	36

Chapter 3	Rule-Based Self-Adaptation in Photonic NoCs for Loss-Aware Co-Management of Laser Power and Performance	37
3.1	Introduction	37
3.2	Fundamentals of Photonic NoCs (PNoCs)	38
3.2.1	Physical-Layer Architecture and Operation of PNoCs	38
3.2.2	Power-Reliability-Performance Tradeoffs in PNoCs	40
3.2.3	Modeling of PP^{dB} and IL^{dB} as Functions of BR and Q	41
3.2.4	Variation in IL^{dB} for Every Photonic Packet Transfer	42
3.3	Related Work And Motivation	43
3.3.1	Prior Works on Laser Power Management	43
3.3.2	Motivation for Rule-Based Self-Adaptation in PNoCs	44
3.4	Proposed PROTEUS Framework	45
3.4.1	Overview	45
3.4.2	Search Heuristic Based Design-Time Optimization	45
3.4.3	Impact of Varying Q and BR	46
3.5	Implementation Of <i>PROTEUS</i> Framework	47
3.5.1	Dynamic Adaptation of BR	47
3.5.2	Dynamic Tuning of Q	50
3.5.3	Putting All Together with Rule-Based Lookup	52
3.6	Evaluation	53
3.6.1	Evaluation Setup	53
3.6.2	Comparative Analysis Results	55
3.7	Summary	57
Chapter 4	Photonic Reconfigurable Accelerators for Efficient Inference of CNNs with Mixed-Sized Tensors	58
4.1	Introduction	58
4.2	Preliminaries	59
4.2.1	CNNs with Mixed-Sized Tensors	59
4.2.2	Accelerating CNN Tensor Products	61
4.2.3	Related Work on Photonic CNN Accelerators	63
4.3	Classification and Scalability Analysis of TPC Organizations	64
4.3.1	Classification	64
4.3.2	Scalability Analysis	66
4.4	Need for Reconfigurability in TPCs	69
4.4.1	Perils of Fixed VDPE Size (N) in TPCs from Prior Work	70
4.5	MRR-based Reconfigurable TPC Architectures	72
4.5.1	Reconfigurable VDPE: Structure and Layout	72
4.5.2	Reconfigurable VDPE: Operation	73
4.5.3	Design of MRR Comb Switches	77
4.5.4	System Level Implementation	77
4.6	Evaluation	78
4.6.1	Simulation Setup	78
4.6.2	Evaluation Results	79
4.7	Summary	80

Chapter 5	An Optical XNOR-Bitcount Based Accelerator for Efficient Inference of Binary Neural Networks	83
5.1	Introduction	83
5.2	Preliminaries	84
5.2.1	Binary Neural Networks (BNNs)	84
5.2.2	Processing of BNNs on Hardware	85
5.2.3	Related Work on Optical BNN Accelerators	85
5.3	Our Proposed OXBNN Architecture	87
5.3.1	Overview	87
5.3.2	XNOR-Bitcount Processing Element (XPE)	87
5.4	Scalability Analysis and Mapping	91
5.4.1	Scalability of XNOR-Bitcount Processing Cores (XPCs)	91
5.4.2	Mapping Convolutions on an XPC	92
5.4.3	Latency and Energy Benefits of PCA	95
5.5	Evaluation	95
5.5.1	System-Level Implementation of OXBNN.	95
5.5.2	Simulation Setup	95
5.5.3	Evaluation Results	97
5.6	Summary	98
Chapter 6	A Stochastic Computing Based Optical Accelerator for Ultra-Fast, Energy-Efficient Inference of Integer-Quantized CNNs	99
6.1	Introduction	99
6.2	Preliminaries	101
6.2.1	Convolutional Neural Networks (CNNs)	101
6.2.2	Processing Convolutions on VDPCs	102
6.2.3	Optical Analog VDPC-Based CNN Accelerators	102
6.2.4	Stochastic Computing	103
6.3	Motivation	105
6.3.1	Scalability Limitations of MRR-Based Analog VDPCs	105
6.3.2	Need for Stochastic Computing	106
6.4	Our Proposed SCONNA Architecture	107
6.4.1	Overview of SCONNA VDPC	107
6.4.2	Optical Stochastic Multiplier	109
6.4.3	Photo Charge Accumulator (PCA)	110
6.5	Scalability Analysis of SCONNA Architecture	111
6.5.1	Operating Speed and Latency Overhead of OSM	112
6.5.2	Achievable Size of SCONNA VDPC	112
6.5.3	Accumulation Capacity and Error Susceptibility of PCA	114
6.6	System-Level Implementation and Evaluation	114
6.6.1	System-Level Implementation of SCONNA	114
6.6.2	Simulation Setup	115
6.6.3	Evaluation Results	116
6.6.4	Inference Accuracy Results	118
6.7	Related Work on Optical CNN Accelerators	118

6.8	Summary	119
6.9	Acknowledgments	119
Chapter 7 A Comparative Analysis of Microrings Based Incoherent Photonic GEMM Accelerators 120		
7.1	Introduction	120
7.2	Preliminaries	121
	7.2.1 Processing of CNNs on Hardware Accelerators	121
	7.2.2 Related Work on Optical GEMM Accelerators	121
7.3	Organizations of MRR-based GEMM Accelerators	122
	7.3.1 Description of Various Blocks that Manipulate Optical Channels in MRR-based GEMM Accelerators	123
	7.3.2 Motivation	125
7.4	Circuit-Level Comparative Analysis	126
	7.4.1 Impacts on Power Penalty Due to Crosstalk Effects	127
	7.4.2 Impacts Due to Optical Signal Losses	129
	7.4.3 Scalability Analysis	130
7.5	Evaluation	132
	7.5.1 System Level Implementation	132
	7.5.2 Simulation Setup	132
	7.5.3 Evaluation Results	133
7.6	Summary	135
Chapter 8 A Hybrid Time-Amplitude Analog Optical Accelerator with Flex- ible Dataflows for Energy-Efficient CNN Inference 137		
8.1	Introduction	137
8.2	Preliminaries	138
	8.2.1 Processing of CNNs on Hardware	138
	8.2.2 Related Work on Optical CNN Accelerators	140
	8.2.3 Motivation	142
8.3	Our proposed HEANA Architecture	143
	8.3.1 Overview	143
	8.3.2 HEANA's Dot Product Element (DPE)	143
8.4	Mapping of Different Dataflows on HEANA DPU	152
	8.4.1 Output Stationary Dataflow	154
	8.4.2 Input Stationary Dataflow	157
	8.4.3 Weight Stationary Dataflow	159
	8.4.4 Operational Logistics and Benefits of HEANA	161
8.5	Scalability Analysis	162
8.6	Evaluation	163
	8.6.1 System Level Implementation of HEANA	163
	8.6.2 Simulation Setup	165
	8.6.3 Evaluation Results	166
	8.6.4 Inference Accuracy Results	170
8.7	Discussion	170

8.8	Summary	171
Chapter 9	Stochastic Computing based Optical Accelerator with Functional Reconfigurability for efficient inference of Heterogeneous Quantized CNNs	173
9.1	Introduction	173
9.2	Preliminaries	175
9.2.1	Quantization of Convolutional Neural Networks	175
9.2.2	Processing Quantized CNNs on Hardware	176
9.2.3	Related Work on Optical CNN Accelerators	176
9.2.4	Stochastic Computing	178
9.3	Motivation	178
9.4	Our Proposed Architecture	180
9.4.1	Overview of our SCOAR Processing Unit	180
9.4.2	Operational Modes of SCOAR Processing Element	181
9.4.3	Reconfigurable Logic Gate	183
9.4.4	RLG Input Peripherals	184
9.4.5	Balanced Photocharge Accumulator	185
9.5	Scalability Analysis	186
9.6	Evaluation	187
9.6.1	System Level Implementation of SCOAR	187
9.6.2	Simulation Setup	188
9.6.3	Evaluation Results	190
9.6.4	Inference Accuracy Results	191
9.7	Summary	192
Chapter 10	Conclusions and Future Work	193
10.1	Future Work	195
	Appendix	198
	Bibliography	199
	Vita	216
	Education	216
	Provisional Patent	216
	Publications	216
	Journal Publications	216
	Conference Publications	216

LIST OF TABLES

2.1	Various Loss And Power Parameters	36
3.1	Various Loss And Power Parameters	54
4.1	List of abbreviations and their full forms used in this Chapter. Definition and values of various parameters (obtained from [22]) used in Eq. 4.1, Eq. 4.2, and Eq. 4.3 for the scalability analysis of AMM and MAM TPCs.	65
4.2	VDPE size (N) at 4-bit precision across various DRs for different TPC architectures.	68
4.3	Kernel tensor shapes (K, K, D), the total number of such kernels (F) and corresponding DKV sizes (S) for EfficientNet_B7[170], as an example CNN with a large number of DSCs. (FC=Fully Connected Layer and other abbreviations are defined in Table 4.1). The K, D, F values were extracted from Keras Applications [37].	71
4.4	Design parameters of various comb switch (CS) designs used in our RMAM and RAMM TPCs for various DRs.	81
4.5	ADC area and power overheads.	81
4.6	Accelerator Peripherals Parameters [144].	81
4.7	VDP Element Parameters [116].	81
4.8	VDPE counts of various accelerators.	82
5.1	Definition and values of various parameters used in Eq. 5.3, Eq. 5.4, and Eq. 5.5 (from [9]) for the scalability analysis. Definitions of PCA parameters γ and α	92
5.2	XPC Size N , PCA bitcount capacity values (γ and α), for different data rates (DRs).	93
5.3	Accelerator Peripherals and XPE Parameters [152]	96
6.1	VDPE size N for input/weight precision={4,6}-bit at data rates (DRs)={1,3,5,10}GS/s, for AMM and MAM VDPCs.	106
6.2	Total number of kernels (T_L) of different DKV sizes (S) for various CNNs. The T_L values were extracted for trained CNN models from Keras Applications [37].	106
6.3	List of abbreviations and their full forms used in this Chapter. Definition and values of various parameters (obtained from [9]) used in Eq. 4.3, Eq. 6.3, and Eq. 6.4 for the scalability analysis of our SCONNA VDPCs.	114
6.4	Peripherals Parameters for Accelerators [6].	115
6.5	Top-1 and Top-5 inference accuracy comparison of SCONNA versus MAM for 8-bit quantized CNNs {GoogleNet (GNet), ResNet50 (RNet50), MobileNet_V2 (MNet_V2), ShuffleNet_V2 (SNet_V2)} and ImageNet dataset [45].	118

7.1	Classification of prior MRR-based analog accelerators based on their DPU organization	125
7.2	Crosstalks effects present in various DPU organizations.	129
7.3	Optical losses present in various DPU organizations.	130
7.4	Definition and values of various parameters used in Eq. 7.1, Eq. 7.2, and Eq. 7.3 (from [9]) for the scalability analysis.	130
7.5	DPU size (N) and DPU Count (#) at 4-bit precision across various DRs for different accelerators architectures.	132
7.6	Accelerator Peripherals and DPU Parameters [152]	133
8.1	Number of analog-to-digital (AtoD) conversions required by various optical DPUs for a GEMM operation. C and D are the height and width of the output matrix, where K is the width (height) of the input (weight) matrix. N=DPE size.	162
8.2	Definition and values of various parameters used in Eq. 8.1, Eq. 8.2, and Eq. 8.3 (from [9, 152, 186]) for the scalability analysis.	164
8.3	DPU size (N) and DPU Count (#) at 4-bit precision across various DRs for different accelerators.	165
8.4	Accelerator Peripherals and DPU Parameters [152]	165
8.5	Top-1 and Top-5 inference accuracy comparison of HEANA versus MAW for 8-bit quantized CNNs {GoogleNet (GNet), ResNet50 (RNet50), MobileNet_V2 (MNet_V2), ShuffleNet_V2 (SNet_V2)} and ImageNet dataset [45].	171
9.1	Definition and values of various parameters used in Eq. 9.1, Eq. 9.2, and Eq. 9.3 (from [9, 152, 186]) for the scalability analysis.	187
9.2	Accelerator Peripherals and DPU Parameters [152]	189
9.3	Top-1 and Top-5 inference accuracy comparison of various binary to stochastic conversation techniques for 8-bit quantized CNNs GoogleNet and ResNet50 and ImageNet dataset [45].	191

LIST OF FIGURES

1.1	An artificial neuron structure analogous to human brain.	2
1.2	A fully connected neural network.	3
1.3	A convolutional Neural Network.	3
1.4	A Recurrent Neural Network.	4
1.5	Number of operations versus top-five error rate for leading ANN designs from ImageNet classification competition. Reproduced from [20].	5
1.6	Performance density (PD) of leading GPU and ASIC platforms. To catch up with the required number of operations, simply increasing the chip area is not feasible. Reproduced from [20].	6
1.7	Performance Density (MAC/s/mm ²) versus Energy Efficiency (J/MAC) of various ANN accelerator platforms. Reproduced from [41].	7
1.8	Illustration of a convolution operation and its equivalent vector dot product operation between Input (6x6) and Kernel (3x3).	8
1.9	Overview of the co-designed PIC based accelerator for CNNs. Reproduced from [34].	9
1.10	A detailed illustration of a co designed PIC based dot product accelerator with FPGA as controller unit, L by K DPUs interconnected with photonic interconnects.	10
1.11	(a) An all pass microring resonator(MRR) (b) An add-drop MRR with radius R, resonance wavelength λ	10
1.12	Illustration the 4×4 swirl topology of the photonics reservoir under investigation. Each node is a nonlinear microring resonator. Reproduced from [39].	12
1.13	(a) The through port transmission of a MRR with pass band and input wavelength operating at λ_{in} (b) MRR pass band being shifted to the left to imprint amplitude value x1 on λ_{in} (c) MRR pass band is shifted further to the left to imprint amplitude value x2 on λ_{in}	12
1.14	A set of cascaded MRRs performing 4x4 dot product operation for vectors x and w.	13
1.15	An on-chip photonic link.	14
1.16	(a) A Reservoir Computer setup (b) A Delay Feedback Reservoir Computing setup. Reproduced from [13]	15
1.17	A MRR based DPU performing M dot product operations of size N	15
1.18	Conceptual breakdown of optical power budget usage and dependency of DPU size N on supported bit precision B for different values of $B=\{2, 3\}$ -bits across datarates DR= $\{1, 5\}$ GS/s.	16
2.1	Schematic of a reservoir computing (RC) accelerator	24
2.2	Illustration of (a) the pre-processing (masking) of input signal, and (b) the generation of delayed feedback reservoir (DFR) states and final output, for a typical DFRC accelerator.	27

2.3	An add-drop microring resonator (MR) with radius R , resonance wavelength λ , and coupling waveguides with cross-coupling coefficients k_1, k_2 and self-coupling coefficients r_1, r_2 [28].	28
2.4	Schematic layout of our proposed MR-based DFRC accelerator. The parts enclosed in the red colored boxes can be integrated on a chip.	29
2.5	NRMSE values for ‘Silicon MR’, ‘All Optical (MZI)’ and ‘Electronic (MG)’ for NARMA10 and Santa Fe timeseries tasks.	33
2.6	SER values of ‘Silicon MR’, ‘All Optical (MZI)’ and ‘Electronic (MG)’ for Nonlinear Channel Equalization task, with SNR ranging from 12dB-32dBs.	34
2.7	Training time of ‘Silicon MR’, ‘All Optical (MZI)’ and ‘Electronic (MG)’ for tasks NARMA10, Santa Fe and Nonlinear Channel Equalization.	35
3.1	Schematic physical-layer layout of the PNoC architecture from [35]. This figure also explains the concepts of packet frame delay and changing insertion loss (IL^{dB}) for every packet transfer	39
3.2	Filter Crosstalk Penalty (PP_{Xtalk}^{Fil}) as a function of quality factor (Q) for various values of signal Bitrate (BR). Evaluation is done using Eq .3.2 and Eq. 3.3 for 0.37nm wavelength spacing and $N_\lambda=55$ at 1550nm operating wavelength	43
3.3	(a) Frame Delay and Optical Power Efficiency for different inser-tion loss values (indicated by different colors) for OPA and <i>PROTEUS</i> (indicated with different shapes); (b) Utilization of P_{Laser} and Aggregated Datarate for OPA and <i>PROTEUS</i> across different insertion loss values.	48
3.4	The schematic of the reconfigurable serializer and deserializer units at the sender and receiver gateway interfaces. These units along with the up-scaled clock rates provided from the clock distri-bution H-tree and switches S_1, S_2, S_3 and S_4 enable dynamic adaptation BR	49
3.5	(a) Two-point coupler arm based MR modulator; (b) Two-point coupler arm based MR filter; (c) Cross-sectional view along AA' of the PN-junction embedded in the coupler arms of the MRs.	51
3.6	Variation of Q (Q -factor) and extinction ratio in our considered MR designs from Fig. 3.5 with applied Q -Tuning Power.	52
3.7	Schematic implementation of our <i>PROTEUS</i> framework on the enhanced Flexishare PNoC architecture from [35].	53
3.8	Total power (electrical laser, thermal tuning, and overhead power) dissipation results for the ABM, OPA, and <i>PROTEUS</i> enabled var-iants of our considered enhanced Flexishare PNoC architecture. Overhead is for adapting Q and BR	55
3.9	Normalized average latency for the ABM, OPA, and <i>PROTEUS</i> enabled variants of our considered enhanced Flexishare PNoC architecture. Results are normalized with respect to the ABM technique.	56
3.10	Aggregate energy-per-bit (EPB) results for the ABM, OPA, and <i>PROTEUS</i> enabled variants of our considered enhanced Flexishare PNoC architecture.	57

4.1	Illustration of various types of convolutions.	61
4.2	Example decomposition of CNN tensors for tensor product acceleration, for (a) standard convolution (SC), and (b) depthwise convolution (DC).	62
4.3	Illustration of common TPC organizations. (a) AMM organization, (b) MAM organization, and (c) Summation Element.	64
4.4	Supported VDPE size N and the optical received power (dBm) for bit precision = {1, 2, 3, 4, 5, 6, 7, 8}bits at data rates (DRs) = {1, 3, 5, 10}GS/s, for MAM TPCs.	68
4.5	Supported VDPE size N and the optical received power (dBm) for bit precision = {1, 2, 3, 4, 5, 6, 7, 8}bits at data rates (DRs) = {1, 3, 5, 10}GS/s, for AMM TPCs.	69
4.6	VDPE utilization % (utilized VDPE area/total area) for MAM (HOLYLIGHT[144],N=44), AMM (DEAPCNN[140],N=31), RAMM (N=31), and RMAM (N=43) at DR=1GS/s and 4-bit precision for various DKV sizes corresponding to DCs and PCs.	72
4.7	Schematic of our invented reconfigurable VDPE, employing a DKV element and one group of comb switch (CS) pairs corresponding to the reconfiguration Mode 2.	73
4.8	Example operation of our reconfigurable VDPE for various cases depending on the S and N values, for $x=9$. Here, ON and OFF CS pairs, respectively, represent Mode 2 and Mode 1 of operation.	74
4.9	System level overview of a CNN accelerator that employs our RMAM/RAMM TPCs.	78
4.10	Area proportionate comparison of FPS for various accelerators across different CNNs and data rates (DRs). Results are normalized with respect to RMAM at 1 GS/s.	82
4.11	Area proportionate comparison of FPS/W for various accelerators across different CNNs and data rates (DRs). Results are normalized with respect to RMAM at 1 GS/s.	82
5.1	(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.	86
5.2	Schematic of an XNOR-Bitcount Processing Core (XPC) of our OXBNN accelerator. Our OXBNN employs binary value set $\{0,1\}$	87
5.3	(a) Schematic of our Optical XNOR Gate (OXG). (b) Spectral operation of OXG. (c) Transient analysis of OXG.	89
5.4	Photo-Charge Accumulator (PCA) Circuit. V_{REF} is the threshold required in the <i>compare()</i> function discussed in Section 5.2.1. Typically, $V_{REF} = 2.5V$ because we consider the dynamic range of TIR to be 5V.	90

5.5	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.	94
5.6	System-level overview of our OXBNN accelerator.	96
5.7	(a) FPS (log scale) (b) FPS/W for OXBNN versus ROBIN and LIGHT-BULB accelerators.	97
6.1	Illustration of a convolution operation.	101
6.2	Illustration of common analog optical VDPC organizations: (a) AMM VDPC, (b) MAM VDPC. (c) Summation Element.	104
6.3	Multiplication between unipolar stochastic numbers I and W	104
6.4	Schematics of (a) Our SCONNA VDPC (b) Photo-Charge Accumulator (PCA) Circuit.	108
6.5	Schematic of our Optical Stochastic Multiplier (OSM).	110
6.6	(a) Schematic of our Optical AND Gate (OAG), (b) operation of OAG, (c) results of OAG's transient analysis.	111
6.7	(a) Bitrate versus FWHM for our OSM/OAG, (b) Our PCA's analog output voltage versus α	113
6.8	System-level overview of our SCONNA CNN accelerator.	115
6.9	(a) FPS (Log Scale) (b) FPS/W (c) FPS/W/ mm^2 for SCONNA versus MAM and AMM accelerators for $B=8$ -bits.	117
7.1	Convolution operation at a convolution layer with two weight filters and one input feature map (Fmap) having two channels is transformed into a GEMM operation between input matrix \mathbf{I} and weight matrix \mathbf{W}	122
7.2	(a) Common optical signal manipulation blocks found in optical DPUs. Illustration of common incoherent photonic DPU organizations; (b) AMSW DPU, (c) MASW DPU, and (d) SMWA DPU.	123
7.3	Conceptual breakdown of optical power budget usage and dependency of DPU size N on supported bit precision B for different values of $B=\{2, 3\}$ -bits across datarates DR= $\{1, 5\}$ GS/s.	126
7.4	(a) Types of losses and power penalties at different optical signal manipulation blocks of optical DPUs. Illustration of (b) Inter-Modulation crosstalk at MRM input arrays [123, 86], and (c) Filter crosstalk and signal truncation at filters [18].	128
7.5	Supported DPU size N ($=M$) for bit precision= $\{1, 2, 3, 4, 5, 6, 7, 8\}$ bits at data rates (DRs)= $\{1, 5, 10\}$ GS/s, for AMW, MAW, and MWA DPUs.	131
7.6	System-level overview of Photonic GEMM accelerator.	132
7.7	(a) Normalized FPS (log scale) (b) Normalized FPS/W (log scale) (c) Normalized FPS/W/ mm^2 (log scale) for AMW, MAW, and MWA accelerators with input batch size=1. Results of FPS, FPS/W, FPS/W/ mm^2 are normalized with respect to AMW executing ResNet50 at 10 GS/s.	135

8.1	Comparison of CNN dataflow schemes: (a) Output Stationary (b) Input Stationary (c) Weight Stationary. Table reports the buffer accesses required by DPU to process layer 5 of GoogleNet[162].	139
8.2	Illustration of common analog optical DPU organizations.(a) AMW DPU (b) MAW DPU.	139
8.3	Schematic of the Dot Product Unit (DPU) of our HEANA accelerator. .	144
8.4	(a) Structure of our microring modulator (MRM) based hybrid time-amplitude analog optical modulator (TAOM) connected to a balanced photocharge accumulator (BPCA) and (b) representation of analog signals (optical and electrical) at different stages of TAOM.	145
8.5	HEANA DPE consisting of two spectrally hitless TAOMs, connected to our BPCA circuit. The inset showcases analog representations of signals (both optical and electrical) at various stages of our DPE	149
8.6	Colormap plots that depict the (a) accuracy and (b) precision of our TAOM for different values of input optical power, sample rate and step size/time interval between the time-analog signals	151
8.7	Loops involved in GEMM operation between input \mathbf{I} and weight \mathbf{W} , mapped onto a photonic DPU, with (a) Output Stationary Dataflow (b) Input Stationary Dataflow (c) Weight Stationary Dataflow. DPU consists of 2 DPEs ($M=2$), and each DPE can perform dot product operation of size $N=2$. For output stationary dataflow and input stationary dataflow the loops are unrolling to show the computations involved in the evaluation of output \mathbf{O} row 1 whereas for weight stationary dataflow evaluation of output \mathbf{O} column 1 is shown. Note that each DPU call performs two dot product operations simultaneously employing the two DPEs. Scheduling order of computation frames for the last three iterations of outermost loops are omitted for brevity	153
8.8	Mapping of GEMM operation between \mathbf{I} and \mathbf{W} with output stationary dataflow on HEANA DPU compared to AMW DPU. (a) Temporal and Spatial tiling of \mathbf{I} , \mathbf{W} , and \mathbf{O} depending on DPU parameters \mathbf{M} and \mathbf{N} (b) Evaluation of output row 1 (O_1) by HEANA DPU with BPCA (c) Evaluation of output row 1 (O_1) by AMW DPU without BPCA.	156
8.9	Mapping of GEMM operation between \mathbf{I} and \mathbf{W} with input stationary dataflow on HEANA DPU compared to AMW DPU. (a) Temporal and Spatial tiling of \mathbf{I} , \mathbf{W} , and \mathbf{O} depending on DPU parameters \mathbf{M} and \mathbf{N} (b) Evaluation of output row 1 (O_1) by HEANA DPU with BPCA (c) Evaluation of output row 1 (O_1) by AMW DPU without BPCA.	159
8.10	Mapping of GEMM operation between \mathbf{I} and \mathbf{W} with weight stationary dataflow on HEANA DPU compared to AMW DPU. (a) Temporal and Spatial tiling of \mathbf{I} , \mathbf{W} , and \mathbf{O} depending on DPU parameters \mathbf{M} and \mathbf{N} (b) Evaluation of output column 1 (O^1) by HEANA DPU with BPCA (c) Evaluation of output column 1 (O^1) by AMW DPU without BPCA. . . .	161
8.11	Supported DPU size $N (=M)$ for bit precision= $\{1, 2, 3, 4, 5, 6, 7, 8\}$ bits at data rates (DRs)= $\{1, 5, 10\}$ GS/s, for AMW, MAW, and HEANA DPUs.	163
8.12	System-level overview of our HEANA accelerator.	164

8.13	(a) Normalized FPS (log scale) (b) Normalized FPS/W for HEANA versus AMW and MAW accelerators with input batch size=1. Results of FPS and FPS/W are normalized with respect to AMW executing weight stationary dataflow (AMW-WS) for ResNet50 at 10 GS/s.	166
8.14	(a) Normalized FPS (log scale) (b) Normalized FPS/W for HEANA versus AMW and MAW accelerators with input batch size=256. FPS and FPS/W results are normalized with respect to AMW executing weight stationary dataflow (AMW-WS) for ResNet50 at 10 GS/s.	167
8.15	(a) Normalized FPS (log scale), (b) Normalized FPS/W for HEANA versus BPCA-integrated versions of AMW and MAW accelerators with input batch size=1. Results of FPS and FPS/W are normalized with respect to AMW executing input stationary dataflow (AMW _{BPCA} -WS) for ResNet50 at 10 GS/s.	168
8.16	(a) Normalized FPS (log scale), (b) Normalized FPS/W for HEANA versus BPCA-integrated versions of AMW and MAW accelerators with input batch size=256. Results of FPS and FPS/W are normalized with respect to AMW executing input stationary dataflow (AMW _{BPCA} -WS) for ResNet50 at 10 GS/s.	169
9.1	Convolution operation at a convolution layer with two weight filters and one input feature map (Fmap) having two channels is transformed into a GEMM operation between input matrix \mathbf{I} and weight matrix \mathbf{W}	177
9.2	Multiplication between unipolar stochastic numbers I and W	178
9.3	Schematics of (a) Our SCOAR-Stochastic Computing based Optical Accelerator with functional reconfigurability (b) Pheriperals of reconfigurable logic gate (c) Balanced Photocharge Accumulator.	180
9.4	(a)Schematic of our Optical reconfigurable logical Gate (RLG) (b) operation of RLG as XNOR gate (c) operation of RLG as AND gate (d) results of RLGs transient analysis	182
9.5	Operation of SCOAR PE (a) Dot product operation mode (b) XNOR-Bitcount operation mode.	183
9.6	System-level overview of our SCOAR CNN accelerator.	188
9.7	(a) Normalized FPS (log scale) (b) Normalized FPS/W (log scale) for SCOAR versus prior optical accelerators with input batch size=1. FPS and FPS/W results of heterogeneous CNNs are normalized relative to ROBIN'EO for ResNet50, and homogeneous CNNs are normalized relative to ROBIN'EO for VGG-small.	189

Chapter 1 Introduction

Artificial intelligence, especially deep learning, has revolutionized the implementation of various real-world computing tasks such as image recognition, language translation, and problems in autonomous cars [96, 128, 118, 92], due to its high inference accuracy. Artificial Neural Networks (ANNs) are the building blocks of deep learning, which attempt to mimic biological neural networks. However, these ANNs require high computational effort, and to achieve higher accuracy, the model parameter count has been increasing [198]. The trend of network size is growing exponentially for these ANNs, leading to an exponential increment of computational effort. Traditional central processing units (CPUs) cannot meet this computational demand, and hence various electronic hardware accelerators have been designed for processing ANNs. These accelerators include graphical processing units (GPUs), application-specific integrated circuits (ASICs), field-programmable gate arrays (FPGAs) [198], as well as other specialized electronic solutions (e.g., IBM TrueNorth [8], Google TPU [81], Graphcore [57] and Cerebras [30]). However, these electronic accelerators struggle to meet the escalating energy efficiency and performance demands of modern highly scaled ANN models due to the dwindling scaling of CMOS technology as it approaches its physical scalability limit [176]. The demand to meet these ANN computation efforts motivates the need for a new platform for computing, and Photonic Integrating Circuits (PIC) based Accelerators are one such platform. This chapter briefly explains the types of ANNs, the need for accelerators, and how PIC based Accelerators can be used for the computation of ANNs. In addition, this chapter also briefly describes the role of network subsystems in accelerators and the advantages of Photonic Network subsystems over Electrical Network subsystems. Furthermore, this chapter also discusses the challenges faced by PIC based Accelerators and outlines the various solutions proposed to overcome these challenges.

1.1 Artificial Neural Networks

Artificial Neural Networks (ANNs) are a subset of artificial intelligence (AI) aimed at solving tasks with capabilities comparable to humans. These networks draw inspiration from the structure and function of the human brain, seeking to replicate the behavior of biological neurons. Fig. 1.1 illustrates a single artificial neuron and its resemblance to a human neuron. An artificial neuron processes multiple inputs, denoted as x_0, x_1, x_2 , with each input having an associated weight, represented as w_0, w_1, w_2 . The output y_i of the neuron is determined by Equation 1.1, where f represents the non-linear activation function, and b denotes the bias term [96].

$$y_i = f\left(\sum_i (w_i x_i) + b\right) \quad (1.1)$$

These neurons are interconnected to form different layers, including an input layer, multiple hidden layers, and an output layer. Input data is initially fed into the neural

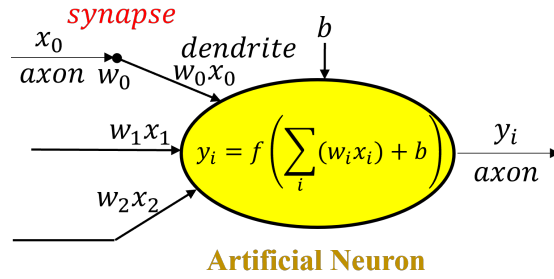


Figure 1.1: An artificial neuron structure analogous to human brain.

network through the input layer. The hidden layers then manipulate or process this data to accomplish the task at hand, with the final output being generated at the output layer. The three widely used types of artificial neural networks (ANNs) are:

- Feed Forward Neural Networks (FNN)
- Recurrent Neural Networks (RNNs)
- Convolutional Neural Networks (CNNs)

1.1.1 Feed Forward Neural Networks

Feedforward Neural Networks (FNNs), also known as Fully Connected Neural Networks, exhibit dense connectivity where each neuron in a layer is connected to every neuron in the subsequent layer. Although FNNs were initially prominent, their dense connections have led to their integration within Convolutional Neural Networks (CNNs) or Recurrent Neural Networks (RNNs), often referred to as dense layers. Figure 1.2 depicts an FNN, typically featuring an input layer that receives input data such as images or audio sequences. Multiple hidden layers within the network manipulate the data to generate the desired output at the output layer. The output of these networks commonly represents probabilities associated with potential outcomes. Each layer's output in FNNs can be described using Equation 1.1, where W_i denotes a weight matrix, with each entry corresponding to the connection strength between layers.

1.1.2 Convolutional Neural Networks

Convolution Neural Networks are mainly used for image classification taking input images to abstract information like the visual cortex system to classify an image with unprecedented accuracy. Typically, a CNN comprises multiple convolutional layers, pooling layers, and a few fully connected layers, as depicted in Figure 1.3. Convolutional layers employ convolution operations to process input images. A kernel of size $k \times k$ traverses the image, generating a single output image at each step. This kernel moves across the input based on a Stride (S) parameter, and optionally,

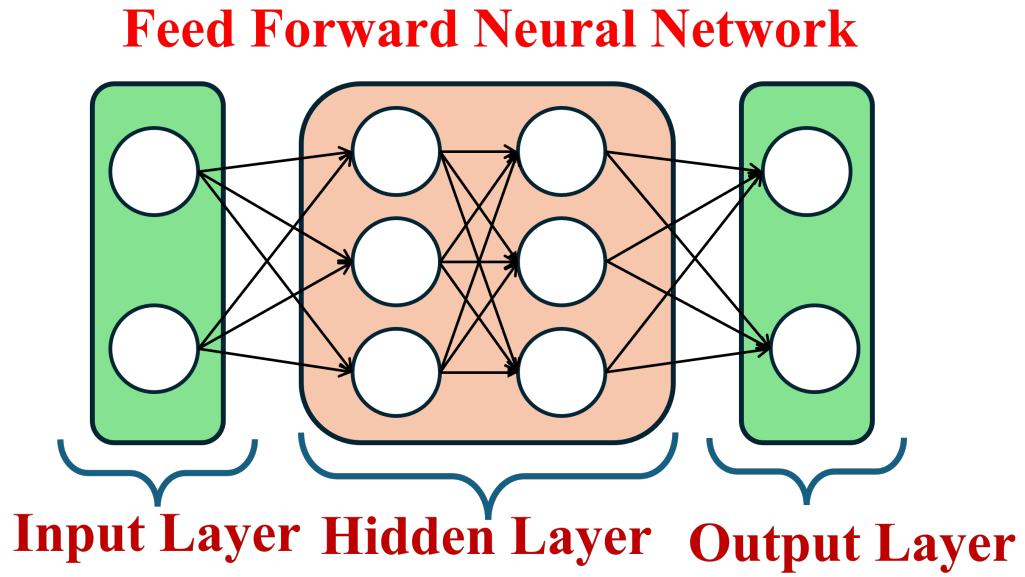


Figure 1.2: A fully connected neural network.

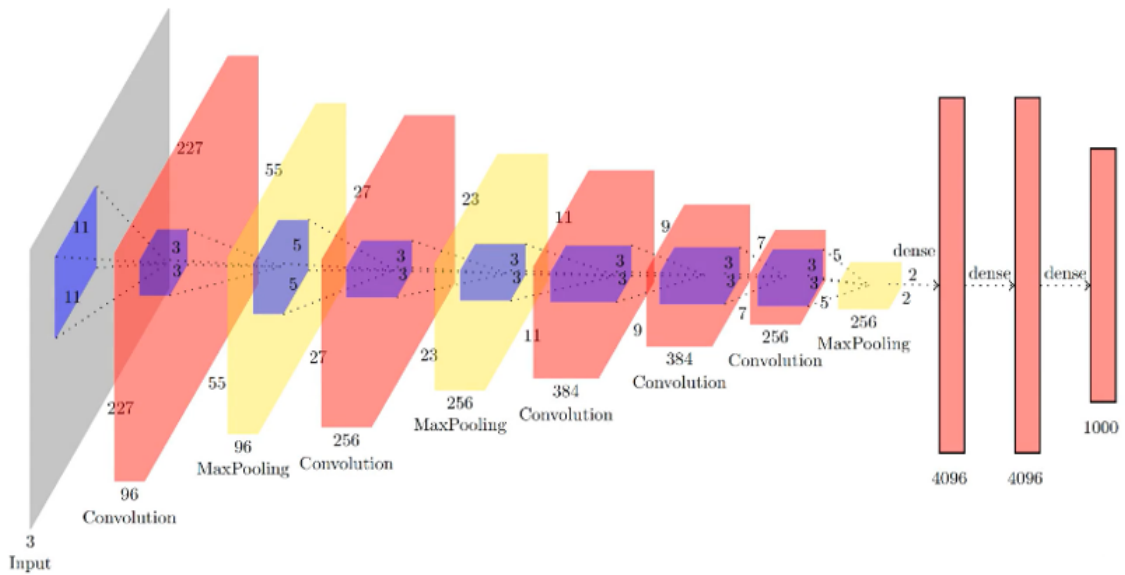


Figure 1.3: A convolutional Neural Network.

empty edges are added to the input based on the Padding (P) parameter. The output dimensions are determined by the input and kernel dimensions, along with the Stride and Padding parameters. A convolutional layer may employ multiple kernels, known as channels. Following a convolutional layer, there may or may not be a pooling layer, which serves to reduce the spatial size of the representation. Towards the end of the CNN architecture, a few dense layers are employed to produce the final output.

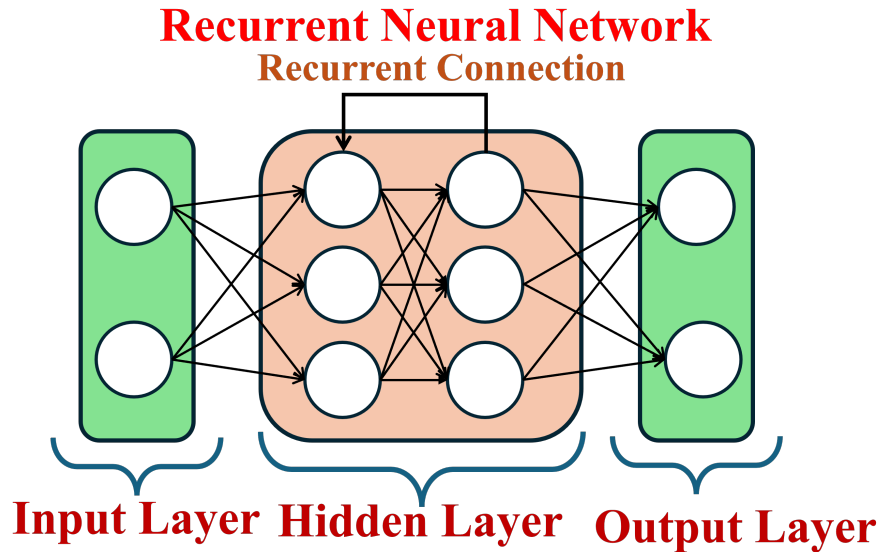


Figure 1.4: A Recurrent Neural Network.

1.1.3 Recurrent Neural Networks

Recurrent Neural Networks (RNNs) find extensive application in processing sequential or temporal data, including tasks such as speech recognition, natural language processing, and time series prediction. A basic RNN architecture resembles a Feedforward Neural Network (FNN), as illustrated in Figure 1.4, but incorporates recurrent connections. In sequential or temporal data, the current output is not solely dependent on the present input but also on preceding inputs or outputs. These recurrent connections facilitate feedback within the system, ensuring that past information remains accessible within the network. Long Short-Term Memory (LSTM) [68] and Gated Recurrent Unit (GRU) [38] are among the most commonly utilized recurrent layers in RNNs.

1.2 Need for ANN Accelerators

Figure 1.5 depicts the relationship between the number of operations performed by state-of-the-art artificial neural network (ANN) models and their achievable top-5 error rates on the ImageNet dataset. Notably, there's a clear trend: as computational efforts increase exponentially, there's a linear improvement in accuracy. This pattern underscores the substantial computational demands associated with enhancing model performance, a trend expected to persist in the coming years. However, traditional CPUs fall short in meeting this computational demand, highlighting the necessity for specialized hardware capable of handling the inference and training tasks of these ANNs. As a result, both industry and academia have proposed various electronics-based accelerators like GPUs, ASICs, and FPGAs [59]. ASIC-based accelerators for ANNs have demonstrated improved performance and energy efficiency

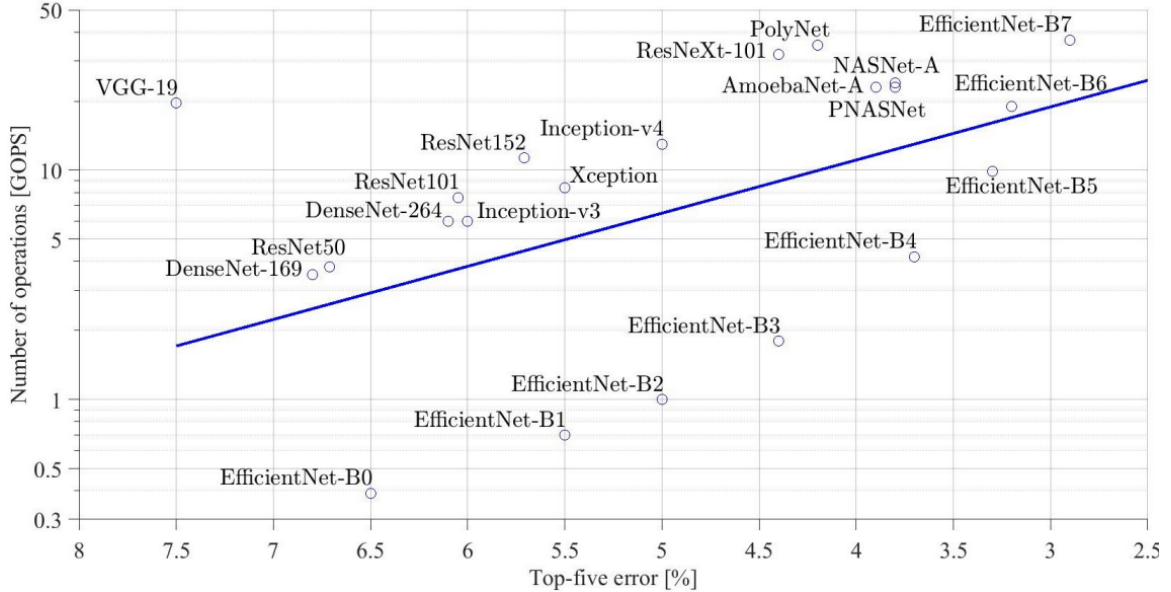


Figure 1.5: Number of operations versus top-five error rate for leading ANN designs from ImageNet classification competition. Reproduced from [20].

over conventional CPU and GPU architectures [59]. Nonetheless, these accelerators may face challenges in meeting the future requirements of ANNs, given the slowdown of Moore’s Law, which limits the computational and performance-to-watt ratio capabilities of emerging electronic processors. Additionally, a significant concern with these accelerators lies in their use of metallic interconnects for data movement, which can suffer from scalability issues, leading to bandwidth, latency, and energy inefficiencies in modern-day processors. Figure 1.6 illustrates the performance density of various ASIC and FPGA-based accelerators. It becomes evident that the number of operations required to compute a neural network increases at a faster rate than hardware performance can keep up with. In pursuit of better power density, GPUs implement specific lower fixed-point arithmetic units. However, the performance density of these accelerators is heavily reliant on the feature size, which is directly linked to Moore’s Law. With the slowdown of Moore’s Law, the feature size cannot be reduced beyond a certain point, impacting computational improvement, as seen in the case of GPUs over the past few years. Therefore, increasing the performance of these electronic-based accelerators to meet the growing demands of ANN computation may prove challenging. Moreover, the operating speed of these accelerators is limited to a range of a few GHz, highlighting the need for a new hardware platform capable of performing ANN computations with higher performance and energy efficiency.

Among the available alternative platforms, Photonic Integrated Circuits (PICs) are promising alternative for the electronic accelerators. Figure 1.7 illustrates the performance density achieved by various ANN accelerator platforms. Notably, photonics-based accelerators stand out for their ability to attain higher performance density

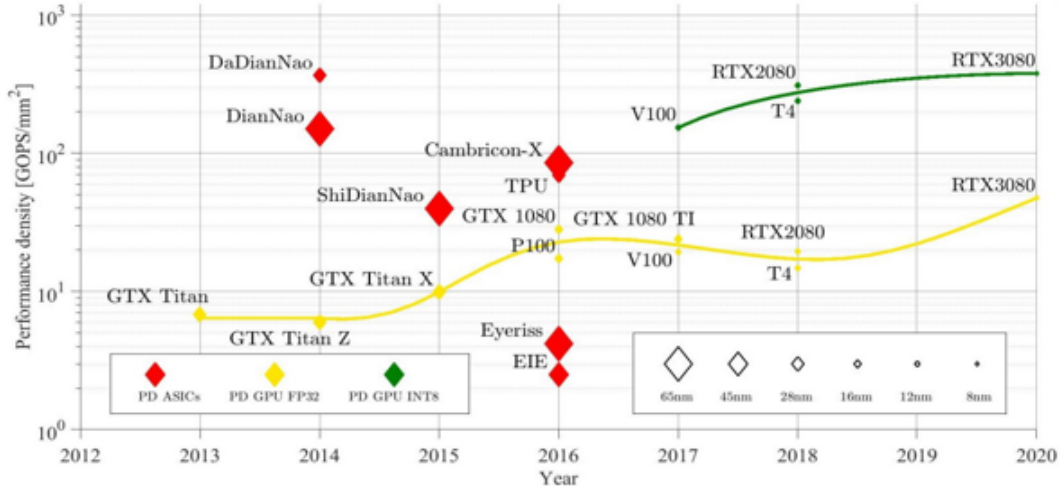


Figure 1.6: Performance density (PD) of leading GPU and ASIC platforms. To catch up with the required number of operations, simply increasing the chip area is not feasible. Reproduced from [20].

compared to electronic-based accelerators, all while maintaining equivalent or even superior energy efficiency. This superiority is attributed to the use of photonic integrated circuits (PICs) in photonics-based accelerators, which enable both computation and communication in the optical domain. PICs offer several advantages, including ultra-low latency and high bandwidth, owing to dense wavelength division multiplexing. Moreover, the integration of PICs with CMOS electronics manufacturing, particularly through silicon photonics, presents a cost-effective solution. Of particular significance is their capability to perform multiply-accumulate (MAC) operations, which are extensively utilized in ANN inference tasks. Additionally, the inherent nonlinearities of PICs present opportunities for their utilization in various AI tasks. All of these advantages position PIC-based accelerators as highly suitable candidates for designing AI accelerators.

1.3 Photonic Integrated Circuits Based AI Accelerators

Photonic Integrated Circuits (PICs) have been widely used in high-speed communication links due to their capability of transmitting data at rates of tens of Gb/s. Additionally, to overcome the metallic interconnect bandwidth and energy bottlenecks, PIC-based photonic interconnects are considered as promising solutions. Photonic links have already displaced metallic links for data transfer at practically every level of the computing hierarchy and are now being investigated for on-chip integration. Remarkably, these PICs can be used to perform matrix-vector multiplication or matrix-matrix multiplication (GEMM) operations that can be leveraged in ANN models like FNNs or CNNs [116], and they also exhibit rich non-linearity that can be employed in Reservoir Computing [140], a less complex counterpart of RNNs. Due to these capabilities, PICs have been used in various computing applications

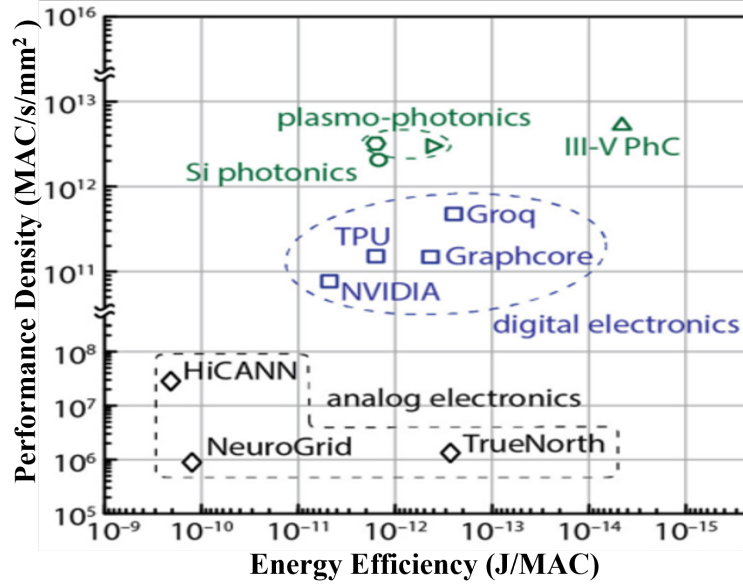


Figure 1.7: Performance Density (MAC/s/mm²) versus Energy Efficiency (J/MAC) of various ANN accelerator platforms. Reproduced from [41].

[163, 144, 72, 22, 143, 131, 142, 53, 11, 39, 89].

Unlike electronic-based accelerators, PIC-based accelerators detect and process information directly in the optical domain, offering not only higher bandwidths, lower latencies, and multiple options for signal multiplexing (wavelength, spatial modes, polarization), but also various non-linearities not present or difficult to attain in electronics [179]. Additionally, PIC-based accelerators address the fan-in and fan-out problems with linear algebra processors as their operational bandwidth can approach the photodetection rate (typically in the hundreds of GHz), which is orders of magnitude higher than electronic systems today that operate at a clock rate of a few GHz [34].

Considering these benefits, PICs have been used for Reservoir Computing, which is utilized for performing temporal tasks. Similar to ANNs, a Reservoir Computer consists of an input layer and an output layer, and instead of hidden layers, it has a reservoir that consists of randomly connected nodes. The weights corresponding to reservoir nodes are randomly assigned and are not trained. Only the weights of output layer nodes are trained, thus reducing the complexity of training. This enables the physical implementation of reservoir computing. PICs such as Mach Zehnder interferometers (MZI), Microring Resonators (MRR), and Semiconductor Optical Amplifiers (SOA) have demonstrated their capability as nonlinear nodes in the reservoir [53, 11, 183, 67, 184, 89, 39, 115].

Furthermore, prior works have extensively focused on designing PIC-based accelerators to accelerate dot product operations, which are the core computing requirement in ANN inference. In CNNs, around 80% of the total processing time is taken by con-

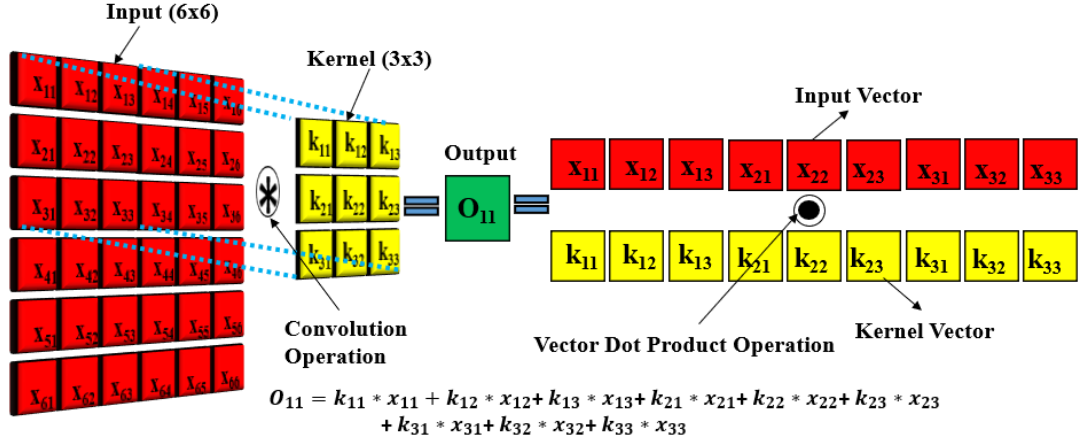


Figure 1.8: Illustration of a convolution operation and its equivalent vector dot product operation between Input (6x6) and Kernel (3x3).

convolution operations [198]. The convolution operation is analogous to a dot product between two vectors. As shown in Figure 1.8, the input and kernel are transformed into vectors X and K. The output of the dot product between these is equivalent to O_{11} . Hence, a dot product accelerator can perform convolutions, and several PIC-based architectures were proposed to perform this operation [169, 211, 201]. Similarly, in FNNs, the dense matrix-vector multiplication can be decomposed into dot product operations, and the dot product core can be used for executing FNNs as well. Any dot product core is composed of multiple dot product units that compute parallel dot product operations. However, to meet the required computation efficiency, co-design of PIC and electronic circuits is necessary. Figure 1.9 illustrates an overview of a co-designed PIC-based accelerator where the PIC-based DPU is only responsible for computing dot product operations, while Electronic Circuitry, generally FPGA, performs all the pre-processing, post-processing, and controlling of the DPU. The pre-processing involves decomposing and mapping for ANN onto each DPU, and post-processing can consist of performing pooling, activation, etc., on the output. In case the dot product operation requires a size greater than the DPU, a partial sum is produced by the DPU, and FPGA is notified using feedback. Each DPU may perform more than one dot product operation simultaneously, and multiple such dot product units controlled by the same FPGA can improve the performance of the accelerator. The interconnection and communication between these multiple DPUs are identical to many-core systems in current processors.

Fig 1.10 illustrates the above-discussed co-design approach in a more detailed manner. In the accelerator, an FPGA loads the ANN model information from the memory and performs pre-processing before calling the DPUs. The digital data is first converted to analog using DAC arrays. The accelerator has k by k DPUs, and the FPGA is responsible for allocating the dot product requests to the DPUs. The

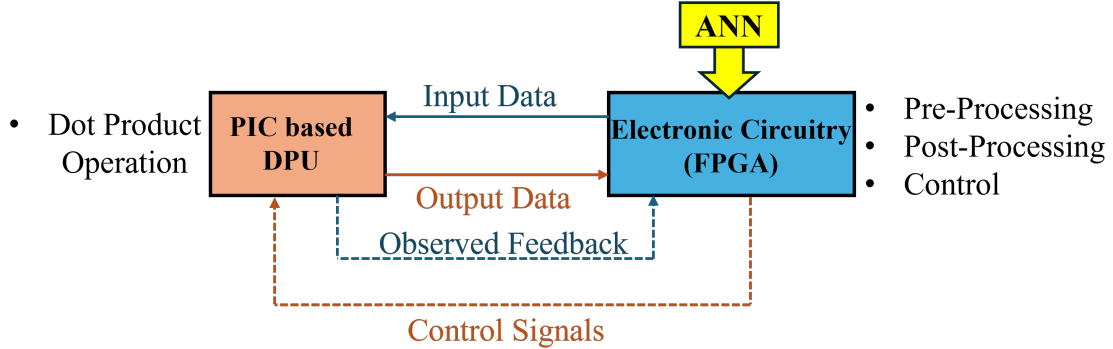


Figure 1.9: Overview of the co-designed PIC based accelerator for CNNs. Reproduced from [34].

dot product units perform the required dot product operations and communicate the results to the FPGA using Photonic Interconnects. To avoid the communication bottlenecks created by electrical interconnects and fully leverage the performance benefits of PIC-based accelerators, photonic interconnects should be employed instead of electrical interconnects. The outputs generated are again converted back to the digital domain using ADC arrays. The FPGA, after obtaining the output values, performs other operations involved in ANN, such as activation, max-pooling, or normalization.

Any PIC-based DPU requires four components to perform computation. (1) To convert the data into an optical domain, for example, modulators (2) Optical data carriers for transporting data in the optical domain to the computation device, for example, waveguides, optical fiber, etc (3) Photonic devices capable of performing computation, for example, MZI, MRR, VCSEL, etc.(4) Photodetectors for converting the data back to the electrical domain. Among the photonic computing devices, MRRs provide better area efficiency, high bandwidth, and low dynamic energy consumption. Therefore, we focus on the challenges of MRR-based accelerators. In the next subsection, we will cover the background of the Microring Resonator and its application in reservoir computing, dot product operation, and photonic network subsystems.

1.3.1 Fundamentals: Microring Resonator

A microring resonator (MRR) is an optical waveguide looped to itself as shown in Fig. 1.11 it is in resonance when the optical length of the resonator is exactly a whole number of wavelengths. Most widely used configurations are all pass MRR (ref Figure. 1.11(a)) and add drop MRR (ref Figure. 1.11(b)). In Add drop configuration, if the incoming wavelength λ_{in} is in resonance, then all the incoming light is coupled into the MRR and no response is observed at throughput. The coupled light is dropped at the drop port. The resonance wavelength of MRR is dependent on the radius (R) and effective refractive index (n_{eff}). The resonance wavelength of the MRR is given

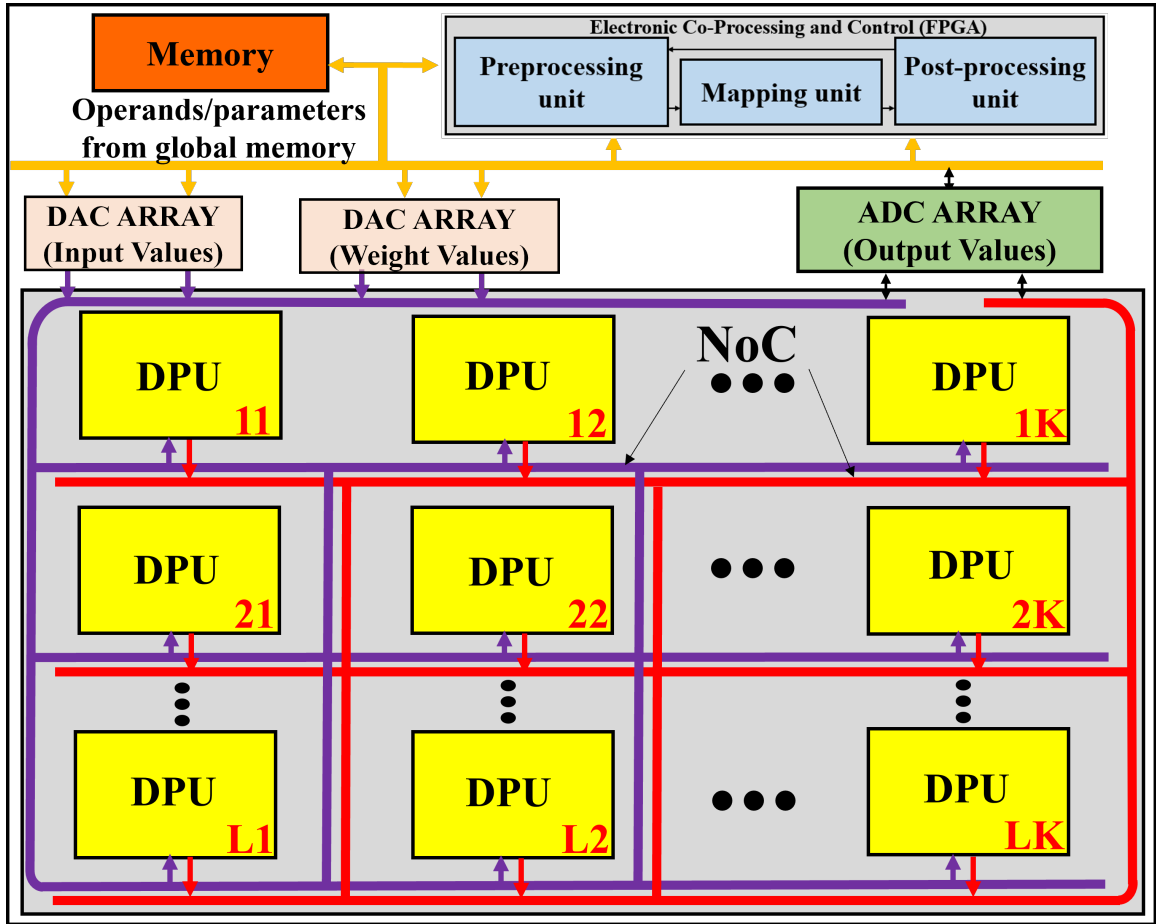


Figure 1.10: A detailed illustration of a co designed PIC based dot product accelerator with FPGA as controller unit, L by K DPUs interconnected with photonic interconnects.

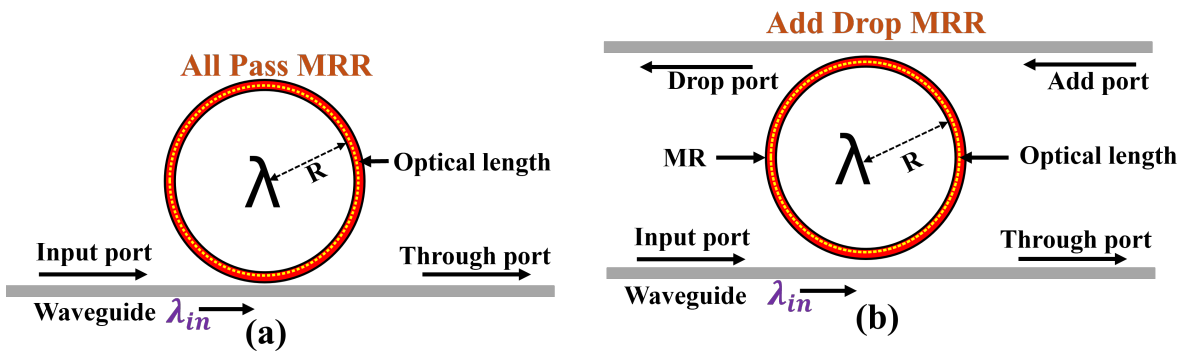


Figure 1.11: (a) An all pass microring resonator(MRR) (b) An add-drop MRR with radius R, resonance wavelength λ .

by the Eq 1.2.

$$\lambda_{res} = \frac{L * n_{eff}}{m} \quad (1.2)$$

where λ_{res} represents the resonance wavelength of the MRR, L denotes the round trip length of the MRR given by $2\pi R$, n_{eff} stands for the effective refractive index of the MRR, and m is an integer. The λ_{res} of the MRR can be altered by changing the R and n_{eff} . The n_{eff} can be modified by using electro-optic tuning [28], which injects or removes charge carriers from the Si core of an MRR. Thermo-optic tuning alters the n_{eff} by changing the temperature of the MRR [28]. MRRs are wavelength division multiplexing (WDM) compatible devices that resonate at a specifically designed wavelength and remain quiescent at all other wavelengths. Below, we discuss the MRR computation capabilities.

1. *Reservoir Computing with MRR*

The MRRs can provide power-dependent nonlinear responses due to effects such as two-photon absorption (TPA) and variation in the nonlinear refractive index [39]. They exhibit a rich nonlinear response at the drop port, making them suitable candidates for reservoir nodes [115]. In prior work [115], a 5×5 reservoir of randomly interconnected MRRs with randomly positioned feedback loops was created, followed by a simple perceptron. This reservoir showcased its computational capability by classifying digital words with a small classification error of 0.1% and 0.5%. Similarly, another MRR-based reservoir [103], shown in Fig. 1.12, was able to perform on par with state-of-the-art counterparts on a nonlinear boolean task under various operating conditions.

2. *Dot Product Operation with MRR*

Figure 1.13(a) illustrates the through-port transmission of the MRR and its passband. The input wavelength at the input port is represented by λ_{in} . To imprint a particular input value x_1 onto λ_{in} , the λ_{res} of the MRR is shifted to the left using electro-optic tuning. Now, λ_{in} overlaps with the passband at x_1 , and the amplitude of λ_{in} is changed to x_1 . Similarly, to imprint a value x_2 , the MRR's λ_{res} is further shifted to the left as shown in Figure 1.13. Thus, a value can be imprinted on λ_{in} by an MRR if the MRR changes the amplitude of λ_{in} to the input value x_1 . On the same waveguide, another MRR operating at this λ_{res} can imprint weight value w_1 onto the x_1 -imprinted λ_{in} , resulting in an amplitude equivalent to $x_1 \cdot w_1$. Since MRRs are WDM-compatible devices, we can cascade multiple MRRs to perform these products and use a photodetector at the end of the waveguide to perform the summation of these products, achieving the dot product operation.

The Figure 1.14 illustrates a 4×4 dot product operation using a set of cascaded MRRs. The first set of MRRs imprints the input vector values i_1, i_2, i_3, i_4 onto the wavelengths $\lambda_1, \lambda_2, \lambda_3, \lambda_4$, respectively. Subsequently, the second set of MRRs imprints the weight vector values w_1, w_2, w_3, w_4 onto the same wavelengths, resulting in products $x_1 \cdot w_1, x_2 \cdot w_2, x_3 \cdot w_3, x_4 \cdot w_4$. These products

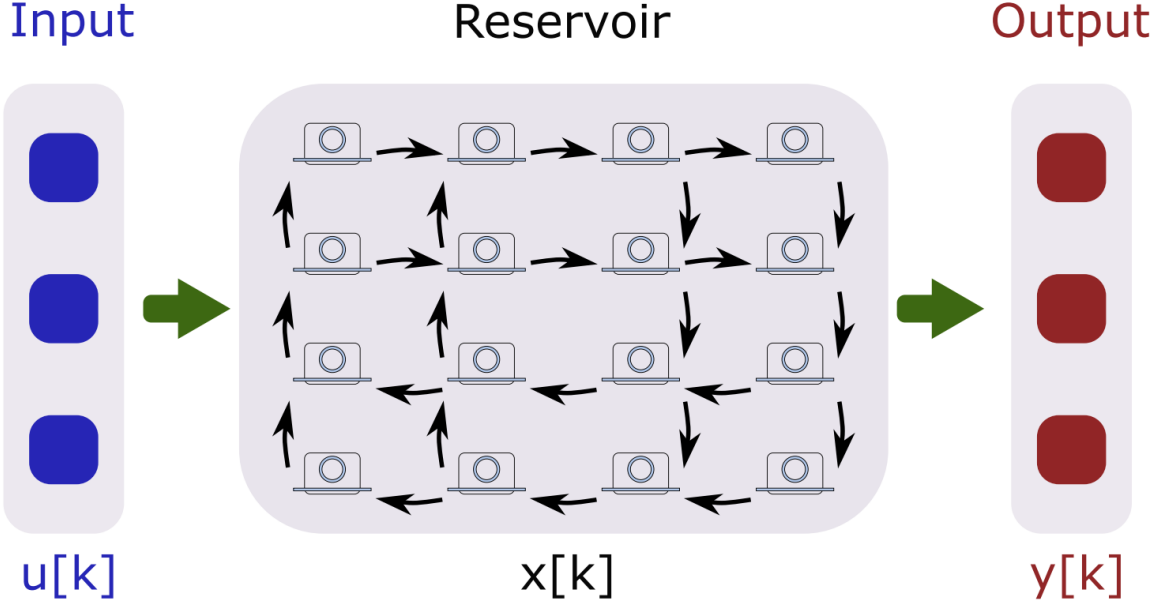


Figure 1.12: Illustration the 4×4 swirl topology of the photonics reservoir under investigation. Each node is a nonlinear microring resonator. Reproduced from [39].

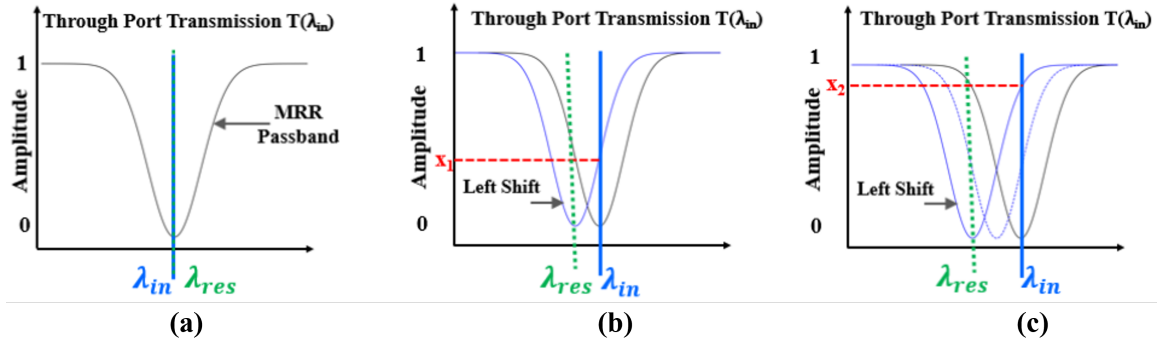


Figure 1.13: (a) The through port transmission of a MRR with pass band and input wavelength operating at λ_{in} (b) MRR pass band being shifted to the left to imprint amplitude value x_1 on λ_{in} (c) MRR pass band is shifted further to the left to imprint amplitude value x_2 on λ_{in} .

are obtained by modulating the amplitudes of the wavelengths $\lambda_1, \lambda_2, \lambda_3, \lambda_4$ accordingly. Finally, at the end of the waveguide, a photodetector performs the summation by generating a photocurrent I_{pd} proportional to the result of the dot product operation between vectors I and W .

3. Photonic Network Subsystems with MRR

As discussed in Section 1.3, an accelerator requires a network subsystem for communication between the various DPUs and the controller. Photonic inter-

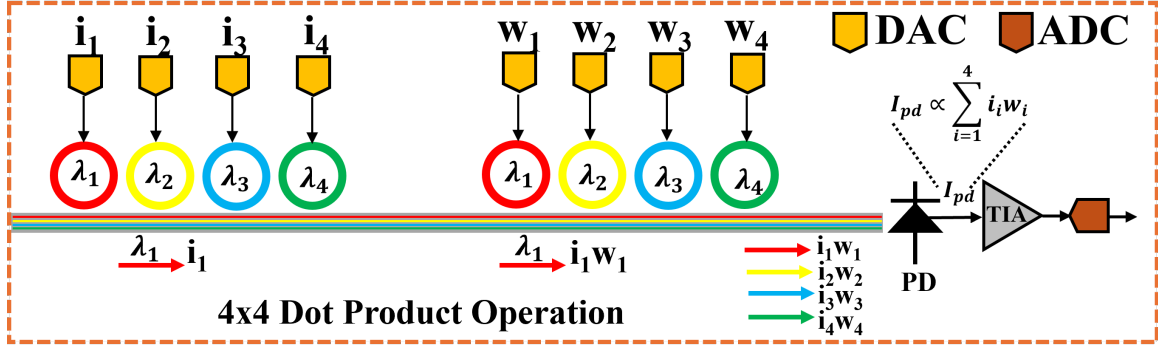


Figure 1.14: A set of cascaded MRRs performing 4x4 dot product operation for vectors x and w .

connects are the optimal choice for carrying out this operation due to their high bandwidth, low latency, better scalability, and low data-dependent energy consumption. The photonic interconnects are composed of photonic links. A photonic link (Figure 1.15) comprises one or more photonic waveguides with dense wavelength division multiplexing (DWDM) of multiple wavelengths into each waveguide.

In a DWDM-enabled waveguide, MRR modulators, arrayed along the waveguide at the source end, modulate input electric signals onto parallel photonic channels. These photonic channels travel through the waveguide and reach the destination end, where an array of MRRs drops the parallel photonic signals onto adjacent photodetectors to recover the electric data signals.

At the transmitter side of the photonic link, each modulator MRR employs a serialization module and a driver circuit capable of producing a sequence of signal bias voltages corresponding to the input sequence of electrical bits. The converted optical data packets are transmitted over different wavelength channels at a higher bit rate compared to the electrical data packets. Therefore, serialization modules are used to enable the conversion between the data rates, and they are implemented using parallel-in serial-out electronic buffers.

Similarly, at the receiver side of the photonic link, each detector MR employs a deserialization module (implemented using serial-in parallel-out electronic buffers) and a transimpedance amplifier (TIA), which amplifies the output signals from the photodetector to digital voltage levels.

1.4 Challenges of PIC based AI Accelerators

1.4.1 Reservoir Computing Challenges

Despite their applicability in reservoir computing, PIC based reservoir computers face below challenge.

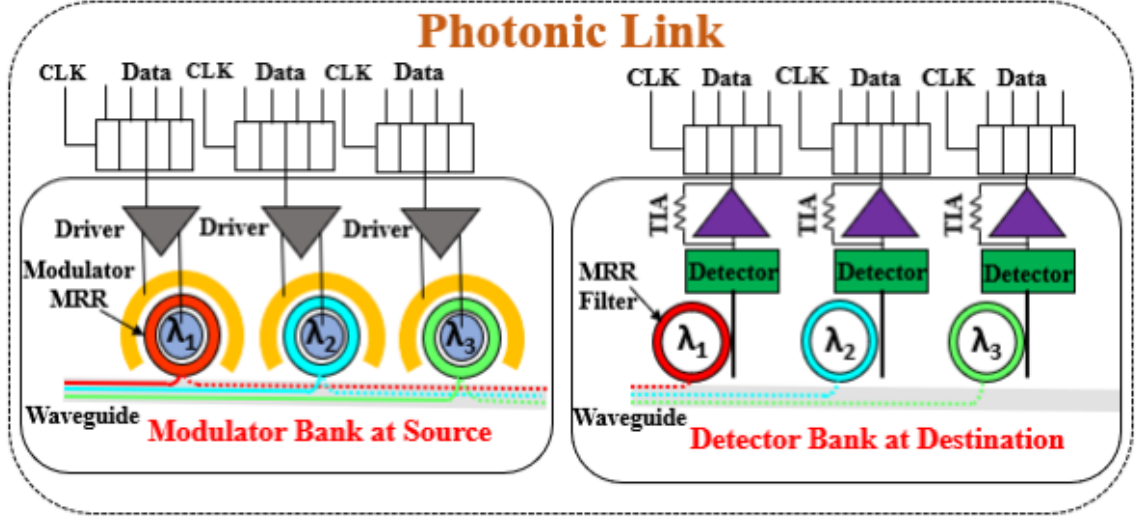


Figure 1.15: An on-chip photonic link.

Scalability and Implementation

Several PIC-based Reservoir Computer (RC) accelerators have been proposed in the past (e.g., [53, 11, 39, 89]). However, these implementations do not scale well, as they may require up to $10^2 - 10^3$ nonlinear nodes per reservoir to achieve acceptable accuracy, posing a significant feasibility challenge. The nodes in these reservoirs are fixed; however, different tasks require a different number of nodes to achieve the best accuracy. To address this shortcoming, an alternative model for RC called delayed feedback reservoir computing (DFRC) has been proposed in [13], which employs a dynamic system consisting of a single NL node subjected to delayed feedback [13]. The traditional RC and DFRC setups are illustrated in Fig 1.16(a) and Fig 1.16(b), respectively. In DFRC, a single NL can act as the desired number of virtual nodes with the help of feedback loops and pre-processing [13]. Prior DFRC accelerators [125, 48, 153] have achieved comparable performance to traditional RC accelerators with low hardware overhead, thereby increasing the ease of implementation and scalability. However, the DFRC accelerators still require long training times and significantly large areas due to the usage of bulky optical fiber spools as feedback loops [125, 48, 153], limiting their applicability to intra-datacenter reservoir computing only. Additionally, these accelerators do not support concurrency in performing multiple RC tasks simultaneously, resulting in reduced throughput. Furthermore, to meet the growing demand for implementing RC-based AI on edge devices (e.g., for applications related to ubiquitous robotics and smart manufacturing), realizing a compact DFRC accelerator that can be fully integrated on a chip is of paramount importance.

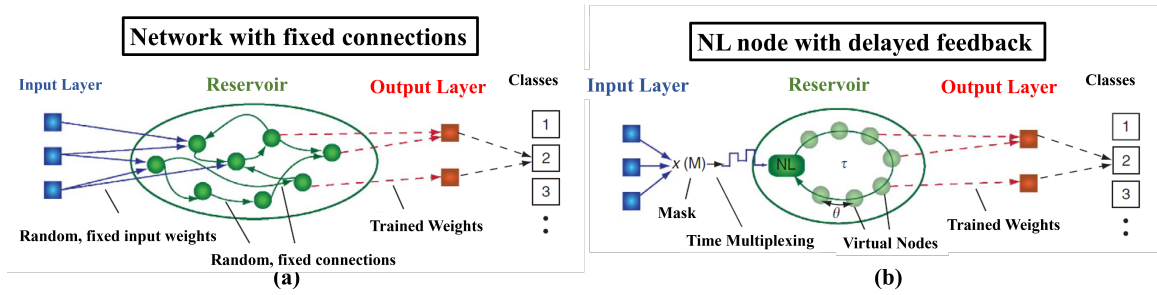


Figure 1.16: (a) A Reservoir Computer setup (b) A Delay Feedback Reservoir Computing setup. Reproduced from [13]

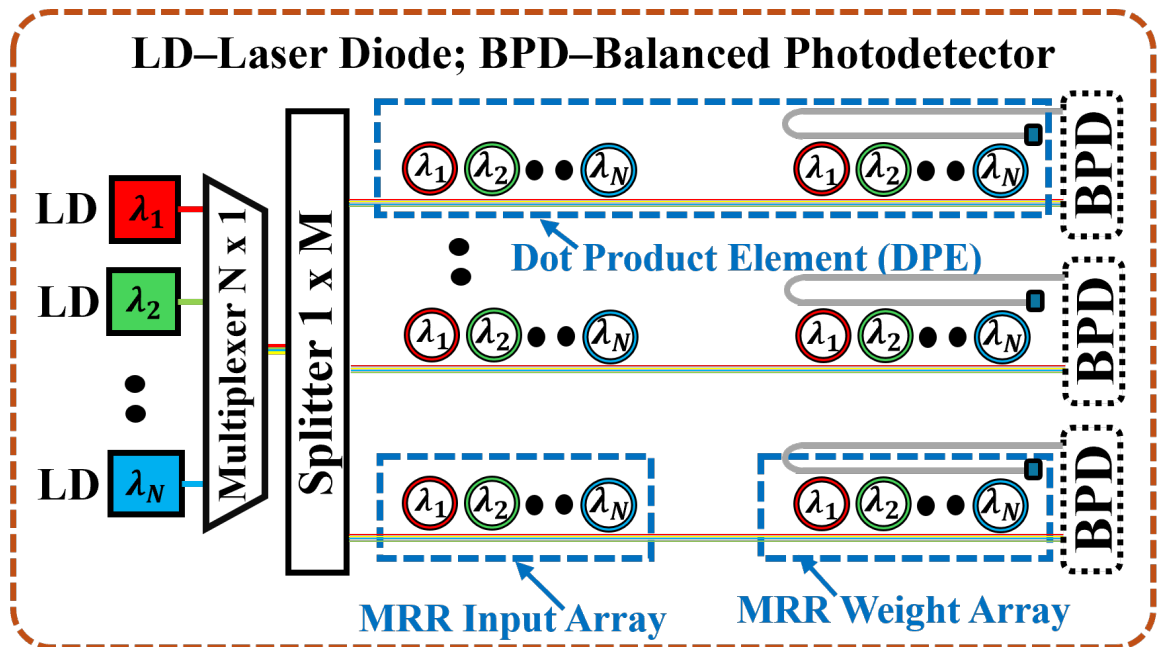


Figure 1.17: A MRR based DPU performing M dot product operations of size N .

1.4.2 Dot Product Acceleration Challenges

Scalability

As discussed earlier, a set of cascaded MRRs on a waveguide can be used to perform dot product operations. The number of MRRs that can be cascaded on a waveguide defines the achievable size N of dot product operation. As shown in Fig. 1.17, such waveguides known as dot product elements (DPEs) are arranged together to form a dot product unit (DPU). The number of such DPEs that can be accommodated gives the M of DPU. The performance achieved by the MRR-based DPUs is largely dependent on three parameters (1) The maximum achievable value of N (fan-in degree/DPE size). Often, the achievable value of N for photonic DPUs is less than the dot product size requirement of dot product operations corresponding to CNN models [152]. In that case, a DPU breaks the dot product into smaller DPU-compatible chunks and generates intermediate results known as partial sums (*psums*). These *psums* are later accumulated using electronic reduction networks [93], to generate the final result. The *psum* reduction latency and energy consumption are non-trivial components of overall latency and energy consumption [120]. Therefore, the value of N plays a crucial role in governing the overall performance of DPUs. (2) The maximum achievable values of M (fan-out degree/count of parallel DPEs). The value of M directly decides the parallelism and consequently achieved throughput by a DPU. (3) The bit precision (B) of input and weight values. If the supported value of B is less than the precision requirement of dot product operations, bit-slicing is applied to input and weight values [156]. Due to bit-slicing, the overall count of dot product operations increases, degrading the throughput and energy efficiency [186]. Therefore, the fundamental driver for achieving high performance from optical DPU lies in maximizing the values of N , M , and B .

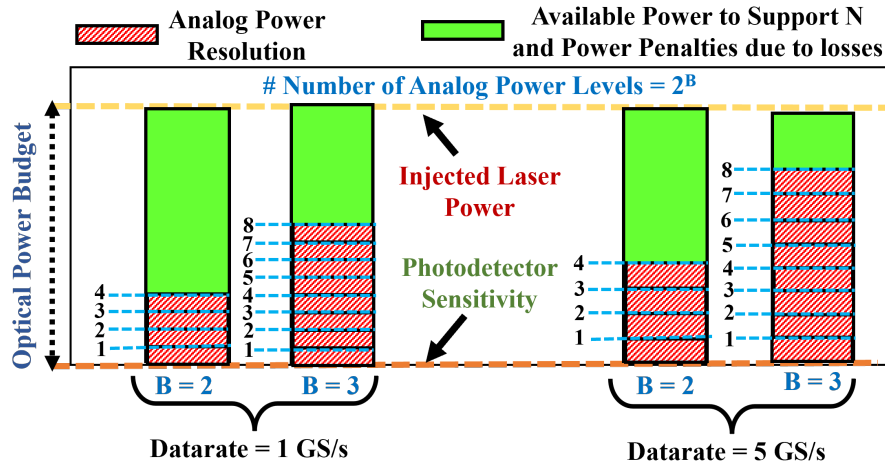


Figure 1.18: Conceptual breakdown of optical power budget usage and dependency of DPU size N on supported bit precision B for different values of $B=\{2, 3\}$ -bits across datarates $DR=\{1, 5\}$ GS/s.

In analog DPUs, a strong trade-off exists among supported values of N , M , and B [9, 152]. The achievable values of M , N , and B also strongly depend on the available optical power budget in the DPUs [9, 152]. This is illustrated in Fig. 1.18, assuming $N=M$, which is a common assumption in the literature [9, 152]. For the bit precision $B=2$, $2^B=4$ analog optical power levels are required that consume a large part of the available power budget, and the remaining power budget is used to support N and power penalty (incurred due to crosstalk effects and signal losses) in the DPU. As B increases to 3-bits, a larger part of the power budget is used to support B , and the available power budget to support N and power penalty further decreases. As a result, the supported value of N decreases too. A similar impact can be observed when the operating data rate of DPUs increases (Fig. 1.18). Low $N=M$ decreases fan-in and fan-out degrees in the DPU, hampering the achievable throughput and energy efficiency.

Reconfigurability

The MRR-based DPU accelerators demonstrated in the literature [22, 109, 157, 156] feature fixed-sized DPEs. However, such fixed-sized DPEs prove inefficient in processing modern CNNs that utilize both standard convolutions and depthwise separable convolutions in their layers, as seen in architectures like Xception [36] and MobileNet_V2 [69]. This inefficiency arises due to the reduced dot product size requirements of depthwise separable convolutions compared to standard convolutions. Because the DPUs have fixed sizes, processing depthwise separable convolution layers often results in low hardware utilization within the DPUs. This low hardware utilization leads to non-amortizable area and static power overheads, while also missing the opportunity to increase processing throughput. Consequently, this diminishes the achievable performance and energy efficiency of such fixed-sized MRR-based DPUs.

Apart from employing different types of convolution layers, modern CNNs also utilize various quantization techniques such as homogeneous quantization, heterogeneous quantization, and binary quantization to create quantized CNNs. Quantized CNNs reduce memory usage and energy requirements by representing weights and inputs with low precision. Among these techniques, heterogeneous quantized CNNs achieve a great balance between accuracy and compression to enhance the speed and efficiency of CNN inference. Heterogeneous quantized CNNs have different precision requirements at different layers; however, existing MRR-based DPUs [160, 22, 157, 215] are designed for executing homogeneous quantized CNNs or binary quantized CNNs. Consequently, they fail to leverage the advantages of heterogeneous quantized CNNs due to their fixed precision support of DPUs. With fixed precision support, these DPUs end up using resources necessary for processing the highest supported precision, thereby diminishing the benefits in latency and power consumption offered by heterogeneous quantized CNNs. Furthermore, binary quantized CNNs leverage XNOR operations for efficient computations using binary values, while homogeneous or heterogeneous quantized CNNs primarily require dot product operations to process quantized integer values with reduced precision. The existing MRR-based DPUs were specifically designed to handle either homogeneous quantized CNNs or binarized

quantized CNNs, but not both simultaneously.

Therefore, due to the lack of reconfigurability in terms of DPU size, precision adaptability, and computing operation, the existing MRR-based DPU accelerators fall short of achieving their potential applicability and achievable throughput and energy efficiency.

Functional Flexibility

The dataflow significantly influences the processing of CNNs on hardware [93]. In CNNs, dataflow refers to how data moves through the network’s layers during computation. This data movement impacts both the efficiency and performance of CNNs on hardware architectures. Optimal dataflow enables the hardware to harness parallelism efficiently, streamline memory accesses to minimize access latency and bandwidth demands, allow the pipelining of operations, and allocate resources effectively to processing units, memory, and interconnects. Overall, optimal dataflow can unlock the full potential of hardware, enabling efficient utilization of resources, reducing latency, and improving overall performance and energy efficiency. However, the optimal data flow varies from one CNN to another. Therefore, for efficient processing of CNNs, the hardware should have the flexibility to support different dataflows efficiently. Unfortunately, the existing MRR-based CNN accelerators [22, 157, 160, 164, 145, 94] lack such flexibility to support different dataflow based on the CNN. In addition, they also lack the functionality of performing in-situ temporal accumulations. This mandates MRR-based CNN accelerators to rely heavily on electronic reduction networks for partial sum reduction. Such reduction involves frequent use of power-hungry ADCs and non-trivial read and write buffer accesses of partial sums.

Implementation Complexity

For current MRR-based DPU architectures [22, 109, 157], the presence of various crosstalk effects and high spectral sensitivity in the MRR weight banks requires the use of extremely complex control procedures for the actuation of weight values [167, 52]. Such control procedures often employ binary search algorithms[167] or need a feedback control circuit [52]. This requirement increases the implementation complexity, mandating the weight actuation control to be separate from the required thermal stability control per MRR. Thus, each MRR requires two feedback control circuits, one for weight actuation and one for thermal stabilization. Similarly, each input MRR already requires a separate input actuation control due to its high-speed operation (typically >1 Gbps). This would increase the number of required feedback control units per weighted optical signal to four because both the input MRR and weighting MRR would require one feedback control unit each for thermal stabilization and another unit each for value actuation. Each control circuit consumes a significant amount of static power [52]. As a result, the generation of each N-sized dot-product at a BPD would increase the total static power consumption by $4N\times$, diminishing the overall energy efficiency of the DPUs.

Area Consumption

The DPEs designed for dot product operation demonstrated in the literature [22, 109, 157] often handle the multiplication of input and weight operands differently. To achieve multiplication, the input operand needs to be modulated onto the incoming optical wavelengths, requiring an additional optical modulator device for weight actuation and multiplication, particularly when utilizing laser sources that provide unmodulated optical power. This necessity of performing multiplication through an additional modulator device introduces an increase in hardware area overhead and complicates operand handling within the DPEs.

1.4.3 Photonic Network Subsystems Challenges

Figure 1.14 illustrates a DPU where each DPE performs multiplications between inputs and weights on different optical wavelength channels and guides these wavelength channels carrying multiplication results to the balanced photodetectors for the dot product operation result or partial sum of the dot product result. In the case of partial sums, the final dot product result is obtained by routing the partial sums to an electronic reduction network consisting of adders. The total optical losses experienced by the optical wavelengths depend on the organization of the photonic network subsystems employed to aggregate the results. Often, to enable reliable transmission, additional optical power, also known as power penalty, is added to each wavelength channel. However, this leads to various power consumption challenges in photonic network subsystems. The challenges can be classified into two categories: the losses challenge and the over-provision challenge. Each of these challenges is discussed below.

High Optical Power Consumption Due to High Optical signal losses

Photonic signals in photonic interconnects experience various types of losses, namely propagation loss, bending loss, splitter and coupling loss, and through loss. Photonic signals propagating inside the waveguide encounter propagation and bending losses. Propagation loss encompasses absorption loss and scattering loss. Non-linear effects in Si, such as Two-Photon Absorption (TPA), induce a strong Free-Carrier Absorption (FCA) effect in silicon [102], significantly increasing absorption losses in waveguides. Si waveguides are also susceptible to high scattering losses due to the sidewall roughness of the waveguides, given the high refractive index contrast between the Si core and SiO₂ cladding. Additionally, splitters and couplers in photonic interconnects incur splitter and coupling losses, while modulators and detectors incur through losses. To ensure that detectors on the receiver end of the photonic interconnect receive sufficient signal power, photonic signals require high laser power. Consequently, high losses result in increased laser power dissipation, negating the energy benefits of photonic interconnects.

High Optical Power Overprovision

For inter-channel crosstalk, the signal power of the same wavelength channel is affected by the noise power from one or more neighboring wavelength channels, while for intra-channel crosstalk, the signal power of a particular wavelength channel is affected by the noise power of the same wavelength channel [173]. The strength of inter-channel crosstalk depends on several factors, namely the quality factor of the MRRs, data rate, and the channel gap between the resonant wavelength of an MRR and its adjacent wavelengths.

High crosstalk in photonic links degrades the optical signal-to-noise ratio (OSNR) and the target bit-error-rate (BER). To compensate for the effects of inter-channel crosstalk and ensure that the target BER remains unaffected, extra optical power is added to each wavelength channel at the transmitter and receiver sides, known as power penalty. This extra optical power is often over-provisioned from the laser source, which can offset the high aggregated data rate, low packet frame delay, and optical power efficiency advantages of photonic interconnects.

1.5 Contributions

Sections 1.4.1, 1.4.3, and 1.4.2 have delineated various design challenges associated with PIC-based AI accelerator systems. In this report, we put forth several solutions to address these challenges and make strides toward designing PIC-based AI accelerator systems, resulting in improved throughput, energy efficiency, scalability, and reconfigurability. The structure/outline of this report, highlighting our contributions, is organized as follows:

In Chapter 2, we address the challenges of scalability and implementation in reservoir computing. We achieve this by presenting a compact architecture for an integrated photonic DFRC accelerator. This accelerator utilizes an active MRR as its nonlinear node and a low-loss photonic waveguide as its feedback loop. By employing an MRR as the nonlinear node, we enable on-chip integration and open the door to concurrent multi-model processing.

In Chapter 3, we tackle the challenge of high optical power over-provisioning in photonic interconnects with a framework that utilizes design-time optimization and runtime self-adaptation techniques to achieve a loss-aware balance between the laser power consumption and performance of photonic network subsystems.

In Chapter 4, we address the underutilization issue caused by the lack of reconfigurability in MRR-enabled DPU-based CNN accelerators. We achieve this in several steps. First, we present a methodology to categorize existing MRR-enabled DPU-based CNN accelerators. Next, we perform a scalability analysis of these DPU categories to understand the relationships between the maximum achievable DPU size, bit precision, and operating data rate. Then, we propose a novel reconfigurable structure for Data Processing Engines (DPEs). These reconfigurable DPEs are then utilized to modify existing DPUs, granting them the capability to dynamically re-aggregate vectors for adaptive resizing of the processed dot product operation. Finally, we eval-

uate the performance of our reconfigurable DPU design and compare it against three different MRR-enabled DPU-based CNN accelerators from prior work.

In Chapter 5, we address the area consumption and implementation complexity challenges of PIC-based binary neural network(BNN) accelerators by presenting a novel MRR-based BNN accelerator. Our design utilizes an array of single-MRR-based optical XNOR gates (OXGs) and highly scalable bit counting circuits called Photo-Charge Accumulators (PCAs). These high-speed OXGs perform XNOR operations with a single MRR, leading to reductions in latency, area, optical losses, and static power consumption. Additionally, our PCA circuits perform in-situ bit count accumulations, eliminating the need for external bit counting circuits and further improving latency and power efficiency. Overall, at system level, our accelerator achieves improved throughput and energy efficiency compared to prior MRR-based BNN accelerators.

In Chapter 6, we address the challenges of scalability in MRR-based CNN accelerators by proposing a merger of stochastic computing and MRR-based CNN accelerators. We break the strong trade-off between the achievable input/weight precision and DPU size for improved scalability. We leverage the innate precision flexibility of stochastic computing by inventing an MRR-based optical stochastic multiplier (OSM). Then, we employ multiple OSMs in a cascaded manner using dense wavelength division multiplexing to forge a novel stochastic computing-based optical neural network accelerator. Our accelerator achieves significantly high throughput and energy efficiency for accelerating inferences of high-precision quantized CNNs.

In Chapter 7, we conduct a comparative analysis of the impact of different photonic interconnect organizations in DPUs on various optical crosstalk effects and signal losses. We determine the scalability limits of each organization and assess their performance in terms of throughput (FPS), energy efficiency (FPS/W), and area efficiency (FPS/W/mm²) for CNN inferences.

In Chapter 8, we address the challenges of implementation complexity and limited functional flexibility in PIC-based CNN accelerators. We achieve this by introducing a novel accelerator design that utilizes a hybrid Time-Amplitude Analog Optical Modulator (TAOM) and a balanced photo-charge accumulator (BPCA). The TAOM leverages a single microring for multiplication, significantly reducing implementation complexity. Our BPCA performs multiple in-situ spatio-temporal accumulations, offering the flexibility to support various dataflows, including input stationary, weight stationary, and output stationary. By seamlessly integrating multiple TAOMs, we create a novel accelerator. We then conduct comprehensive analyses at the device, circuit, and system levels to evaluate its advantages over previous works.

In Chapter 9, we tackle the challenges of reconfigurability, scalability, and limited functional flexibility in PIC-based CNN accelerators by introducing a novel stochastic computing-based optical accelerator with functional reconfigurability. Our accelerator is designed to efficiently execute all types of quantized CNN models, including homogeneous, heterogeneous, and binary quantized CNNs. It utilizes a unique design of reconfigurable single MRR-based logic gates (RLGs), enabling it to adjust its functionality according to the specific requirements of the CNN. Moreover, this work places particular emphasis on the practical implication of integrating on-chip

laser sources. We assess our accelerator throughput, energy efficiency, and inference accuracy in comparison to prior works.

Chapter 10 concludes this report. We recap all our contributions and provide directions for future research.

Chapter 2 Silicon Photonic Microring Based Chip-Scale Accelerator for Delayed Feedback Reservoir Computing

2.1 Introduction

Artificial Neural Networks (ANNs) have achieved remarkable progress in recent years, and they are being aggressively utilized in real-world applications related to artificial intelligence (AI) and machine learning [137]. In general, ANNs mimic biological neural networks. Depending on the type of the computing task, an ANN architecture can be classified as a feedforward network (FNN) used for static or non-temporal data processing [139], or a recurrent neural network (RNN) that is used for dynamic or temporal data processing [64]. Theoretically, RNNs are very powerful tools for solving complex temporal machine learning tasks. But, the application of RNNs to real world problems is not always feasible due to their high computational training cost and slow convergence [64]. To mitigate these shortcomings, Reservoir Computing (RC) was proposed [77], which is a computational framework [191] derived from the RNN models such as the echo state networks (ESNs) [75] and liquid state machines (LSMs) [112].

In an accelerator for RC (Fig. 2.1), data inputs are transformed into spatiotemporal patterns in a high dimensional space using a *reservoir*, and analysis of these patterns is performed by an *output* layer. The inputs are connected to the *reservoir* with weights W_{in} at the input layer, and the *reservoir* is connected to the output layer with weights W_{out} . The *reservoir* consists of large number of nonlinear (NL) nodes (Fig. 2.1) that are connected to each other through the recurrent nonlinear dynamics by weights WR . The output of the *reservoir* is the linear combination of W_{out} and the state of the NL nodes connected to the output layer. The key trait of RC is that the input weights W_{in} and the *reservoir* weights WR are not trained; they are fixed and random. The output weights W_{out} are trained using simple learning algorithms, e.g., linear regression, thus, remarkably reducing the computational cost of learning, compared to the standard RNNs [99]. Thus, an accelerator for RC can have benefits of both the fast information processing and low learning cost [64], compared to RNN accelerators.

Photonics based implementations of RC accelerators are attractive as photonics can have low dynamic power consumption and extremely fast computation. Several photonics based RC accelerators have been proposed in the past (e.g., [53, 11, 183, 67, 184, 89, 39, 115]). However, these implementations do not scale well as they may require up to 10^2 - 10^3 NL nodes per *reservoir*, making their feasibility a big challenge. In contrast, an alternative model for RC called delayed feedback *reservoir* computing (DFRC) has been proposed in [13], which employs a dynamic system consisting of a single NL node subjected to delayed feedback [13]. It is shown in prior work (e.g., [178, 180, 151, 99]) that DFRC accelerators can achieve comparable performance with low hardware overhead compared to the traditional RC accelerators, thereby increasing the ease of implementation and scalability. In fact, prior work also present several

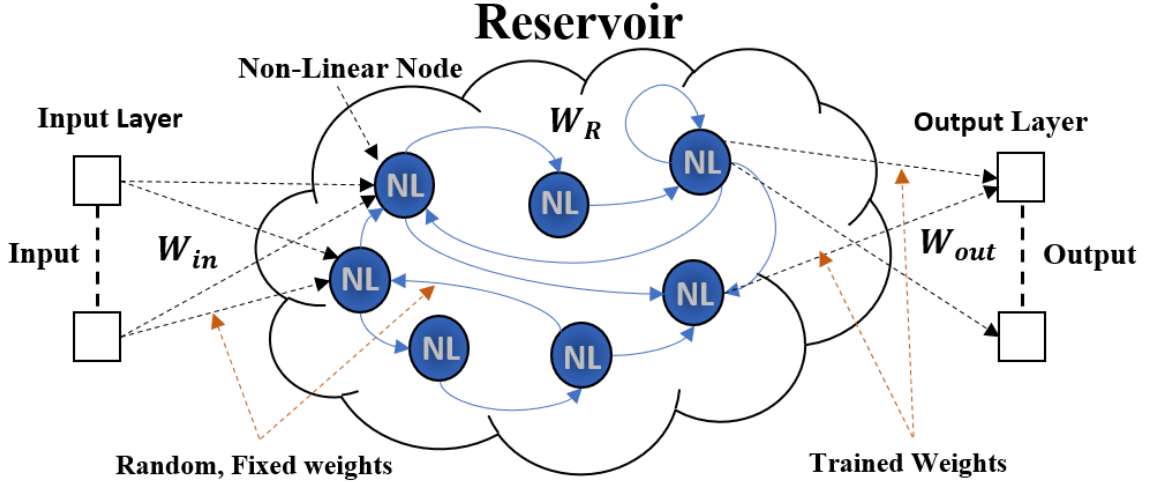


Figure 2.1: Schematic of a reservoir computing (RC) accelerator

photonic DFRC accelerators (e.g., [125, 48, 153]). But unfortunately, the DFRC accelerators from prior work still require long training times and significantly large area, which limits their applicability to intra-datacenter *reservoir* computing only. In contrast, *to meet the growing demand of implementing RC-based AI on the edge devices (e.g., for applications related to ubiquitous robotics and smart manufacturing) (Section II.B), realizing a compact DFRC accelerator that can be fully integrated on a chip is of paramount importance.*

In this Chapter, we present an architecture of a chip-scale photonic DFRC accelerator, which employs a CMOS-compatible active silicon microring resonator (MR) (e.g., [110]) as the NL *reservoir* node and a low-loss on-chip photonic waveguide (e.g., [110]) as the optical feedback loop. Our DFRC accelerator benefits from its MR *reservoir* node’s rich nonlinearity [39, 115] to enable ultra-fast, reasonably accurate, and energy-efficient RC.

Our contributions in this Chapter are summarized below:

- We present a compact architecture of an integrated photonic DFRC accelerator that has an active MR as its NL node and a low-loss photonic waveguide as its feedback loop (Section IV);
- We evaluate our MR-based DFRC accelerator’s efficiency for performing typical RC tasks such as NARMA10 [28], Santa Fe Time Series [28], and Nonlinear Channel Equalization [77];
- We evaluate our MR-based DFRC accelerator in terms of training time and prediction error metrics, and compare it with a photonic [48] and an electronic [13] DFRC accelerators from prior work (Section V).

2.2 Related Work And Motivation

Several accelerators for RC have been reported that use various types of physical systems, substrates, and devices [53, 11, 183, 67, 184, 89, 39, 115, 14, 178, 180, 151]. Some of these accelerators (e.g., [178, 11]) use large reservoirs that employ many interconnected NL nodes (as in Fig. 2.1), whereas the other accelerators (e.g., [48, 125, 153, 185, 140], [151, 99, 46]) use delayed feedback reservoirs (DFRs) with one physical NL node subjected to delayed feedback to behave as N virtual NL nodes (Section 2.3.1). Compared to large reservoirs, DFRs have recently become more popular due to their implementation simplicity. A physical *reservoir* (DFR or large) needs to satisfy a few traits as discussed in [140], to efficiently solve the temporal computing problems. These traits for DFRs mainly depend on the NL nodes. Prior works on DFRC accelerators explore several types of DFR NL nodes, as discussed next.

2.2.1 DFRC Accelerators from Prior Work

In general, a DFRC accelerator employs a physical NL node and a delayed feedback loop. The DFRC accelerators from prior work broadly use either electronic or photonic implementation for the NL node and delayed feedback loop. The electronic DFRC accelerators (e.g., [110, 151, 99]) typically use analog circuits to implement NL transformation. These analog circuits also take care of the required delayed feedback. However, these analog circuits can typically suffer from high capacitive loading, which can significantly reduce the speed and power-efficiency, especially when longer delay is required to accommodate large number of virtual nodes. In contrast, photonic DFRC accelerators (e.g., [48, 125, 153, 185]) enjoy distance-independent and fast computation speed. They employ NL photonic devices, such as Mach-Zehnder Interferometers (MZIs) [125], semiconductor lasers [153], Vertical Cavity Surface Emitting Lasers (VCSEL) [185], as NL nodes. To implement the feedback loop, they typically use a bulky fiber spool [48]. Due to the large size of the photonic NL devices (e.g., a few micro-milli meters for MZIs [62]) and long fiber spools (up to 1.7km [48]), these photonic DFRC accelerators generally yield feedback loop delay (τ) in the micro-milli seconds range, which in turn yields significantly long time (a few tens-hundreds of seconds) for these accelerators to collect the *reservoir* states for output weights training. To decrease the training time, recent work [46] used a deep learning approach with multiple DFRs and fiber spools. However, the use of bulky fiber spools limits the deployment of such DFRC accelerators to high-end computing systems and datacenters only.

2.2.2 Motivation for MR-based Chip-Scale DFRC Accelerators

Due to the emergence of ubiquitous robotics and smart manufacturing, the demand for RC-based information processing and AI is rapidly growing. To this end, bulky fiber spools based photonic DFRC accelerators find limited pertinence. To address this shortcoming, we present a chip-scale DFRC accelerator that uses an active mi-

roring resonator (MR) as the NL *reservoir* node and a low-loss photonic waveguide as the delay feedback loop. In prior work [28] and [76], rich nonlinearity of MR through-port response has been leveraged to realize RC accelerators. However, these MR-based accelerators employ large reservoirs with large number of MR-based NL nodes, and therefore, they do not scale well for integrated applications. In contrast, our proposed DFRC accelerator uses only one active MR as the single physical NL node that can be scaled to N virtual nodes on demand using a photonic waveguide based delayed feedback loop, which makes our DFRC accelerator highly viable for chip-scale applications.

2.3 Background and Fundamentals

2.3.1 Delayed Feedback Reservoir Computing (DFRC)

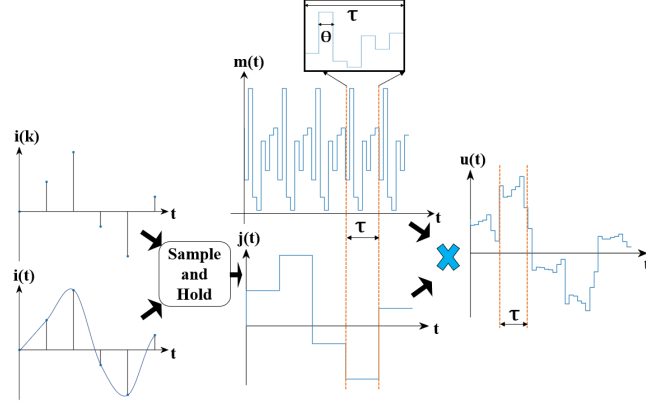
The basic idea of a delayed feedback *reservoir* computing (DFRC) accelerator is to have a single physical NL node to behave as N virtual NL nodes. Fig. 2.2 illustrates the functioning of a typical DFRC accelerator, which consists of (i) pre-processing (masking) of input signals (Fig. 2.2 (a)), (ii) generation of N DFR states (Fig. 2.2 (b)), and (iii) output generation (Fig. 2.2 (b)), as discussed next.

1. Pre-Processing (Masking) of Input Signal

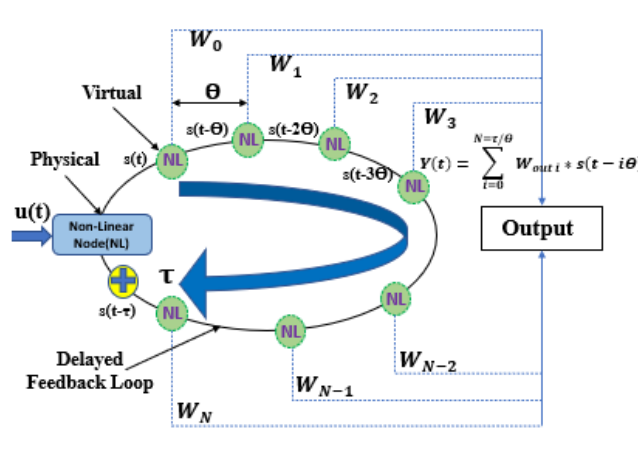
A time-dependent input to the DFRC accelerator can be a continuous-time signal $u(t)$ or a discrete-time signal $u(k)$ (Fig. 2.2(a)). Regardless, such input signal is typically sampled and held to produce a sampled (discretized) continuous signal $j(t)$, with each sample of $j(t)$ being constant for a period of τ . Then, $j(t)$ is multiplied with a periodic masking signal $m(t)$, which plays the role of assigning weights (input weights W_{in} ; Fig. 2.2) to the virtual nodes. The masking signal $m(t)$, with its period being τ , varies its value at each θ interval for total N times during every τ period (Fig. 2.2(a) inset), so that $\tau = N*\theta$. The masking is essential for sequentializing the input, breaking the symmetry of the input and enabling the high dimensional space [140]. The periodic nature of $m(t)$ (i.e., $m(t)$ holds the same value for a corresponding θ in every τ period) ensures that the weight assigned to each virtual node remains identical for all input samples. Multiplying $m(t)$ with $j(t)$ generates the masked signal $u(t)$ that is given as input to the DFR.

2. Generation of Delayed Feedback Reservoir (DFR) states

From Fig. 2.2(b), the physical NL node of the delayed feedback *reservoir* (DFR) nonlinearly transforms $u(t)$ to generate the state of the DFR $s(t)$, and then propagates $s(t)$ along the delayed feedback loop. The state of the DFR is generated at each θ interval, hence producing total N state values (i.e., $s(t)$, $s(t-\theta)$, $s(t-2\theta)$, ..., $s(t-N\theta)$) in each τ period. Each of these N state values represents the output of the corresponding virtual NL node (total N virtual nodes). The input to the physical NL node is the combination of $u(t)$ for current τ period and the DFR response $s(t-\tau)$ (from the feedback loop) that



(a)



(b)

Figure 2.2: Illustration of (a) the pre-processing (masking) of input signal, and (b) the generation of delayed feedback reservoir (DFR) states and final output, for a typical DFRC accelerator.

was obtained for $u(t-\tau)$ of the previous τ period. The governing formula for the DFR states is given in Eq. 2.1- 2.2 [5].

$$s(t - i\tau) = FNL(s(t - \tau), u(t), \theta) \quad 0 \leq i \leq N \quad (2.1)$$

$$u(t) = j(t) * m(t) \quad (2.2)$$

where FNL is the nonlinear transfer function of the physical NL node. The time separation between the two virtual NL nodes is θ . N is the number of virtual nodes and τ corresponds to the total delay of the feedback loop.

3. Training of Output Weights for Final Output Generation

The states of the virtual NL nodes of the DFR are connected to the output

layer with the output weights $W_{out, 0}, W_{out, 1}, \dots, W_{out, N}$ (Fig. 2.2). The final output $Y(t)$ of the DFR is the linear combination of these output weights and the states of the DFR, as given by Eq. 2.3 [5].

$$Y(t) = \sum_{i=0}^{N=\tau\theta} W_{out,i} * s(t - i\theta) \quad (2.3)$$

where $W_{out,i}$ are the output weights, and $s(t-i\theta)$ are the DFR states. The output weights of the DFR are typically trained offline using lightweight algorithms (e.g., linear regression). During training, the output weights are initialized, then the states of the DFR are observed for the input, and the corresponding intermediate output of the DFR is calculated. The deviation in the intermediate DFR output from the correct output (known during training) is referred to as training error. Then, the output weights are optimized using a learning algorithm until the least value of the training error is achieved. Several techniques, such as Least Mean Squares, Recursive Least Mean Squares and Moore-Penrose Pseudo Inverse [24], can be used to converge to optimal output weights during training. In this Chapter, we use the Moore-Penrose Pseudo Inverse technique as it suffers less from converging to local minima.

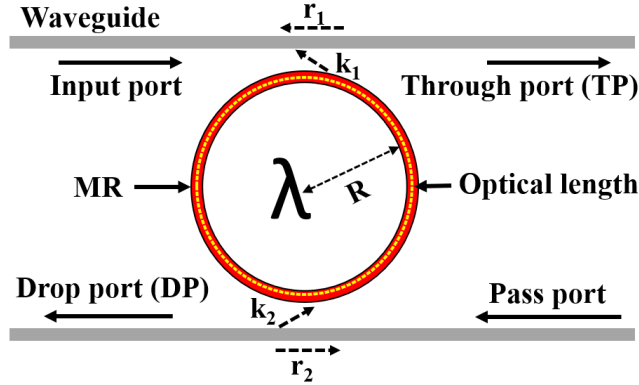


Figure 2.3: An add-drop microring resonator (MR) with radius R , resonance wavelength λ , and coupling waveguides with cross-coupling coefficients k_1, k_2 and self-coupling coefficients r_1, r_2 [28].

2.3.2 Fundamentals: Microring Resonators (MRs)

A microring resonator (MR) is an optical waveguide looped back on itself as shown in Fig. 2.3. It also consists of a coupling mechanism to access the loop. The MR is in resonance when the optical path length inside the MR cavity is an integer multiple of the input wavelength. The incident optical at the input port of the MR is transmitted to the through port and to the drop port (Fig. 2.3). From [76], the drop-port transmission of an active MR shows very rich power-dependent nonlinear response,

due to the two-photon absorption (TPA) effects [76]. This rich nonlinearity of MR’s through-port response has been leveraged in [28] and [76] to realize efficient RC accelerators. In this Chapter, we propose to use this rich nonlinearity to implement the physical NL node of our DFRC accelerator, as discussed next.

2.4 Proposed MR-based Chip-Scale DFRC Accelerator

2.4.1 Overview

Fig. 2.4 illustrates our proposed MR-based DFRC accelerator with photonic waveguide as a feedback loop. The accelerator architecture can be divided into three layers: (i) input layer, (ii) reservoir layer, and (iii) output layer. The input layer consists of a laser source (off-chip) that injects continuous-wave (CW) light of wavelength λ into the on-chip waveguide. The on-chip waveguide is coupled to an MR modulator that modulates the input CW λ with respect to the masked input $u(t)$ to generate a modulated optical signal (optical $u(t)$) in the waveguide at the through port of the MR modulator. From [76], the through port response of an MR modulator does not show significant nonlinearity, therefore, optical $u(t)$ at the through port of the MR modulator in the input layer is not nonlinearly transformed. This optical $u(t)$ propagates along the waveguide and enters the reservoir layer. In the *reservoir* layer,

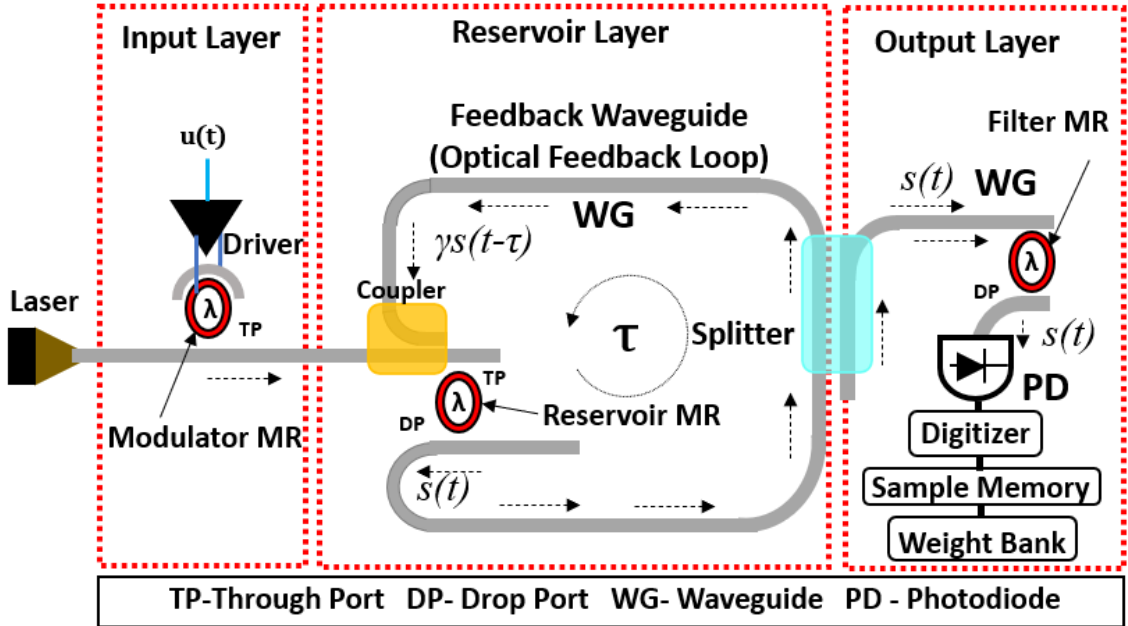


Figure 2.4: Schematic layout of our proposed MR-based DFRC accelerator. The parts enclosed in the red colored boxes can be integrated on a chip.

the input waveguide is coupled with one end of the feedback waveguide (used for delayed feedback loop) via a coupler. This coupler sums the input optical signal $u(t)$

with the feedback optical signal $\gamma s(t-\tau)$, provided from the feedback waveguide that imposes the propagation delay of τ . Just like $u(t)$ (Section 2.3.1), the feedback optical signal $\gamma s(t-\tau)$ also changes every θ interval, and therefore, the summed optical signal $(u(t)+\gamma s(t-\tau))$ at the coupler also changes every θ interval. This $(u(t)+\gamma s(t-\tau))$ signal then passes through the *reservoir* MR that acts as the NL node in the *reservoir*. The drop port of the *reservoir* MR is coupled with the other end of the feedback waveguide. At the drop port of the *reservoir* MR (NL node), the $(u(t)+\gamma s(t-\tau))$ signal is nonlinearly transformed into $s(t)$, which is then injected into the feedback waveguide. Just like $(u(t)+\gamma s(t-\tau))$, $s(t)$ also changes every θ interval during the feedback period τ . The value of $s(t)$ during every θ interval corresponds to the state of a virtual NL node. Thus, to have N virtual NL nodes in the system, total N θ -intervals should be accommodated in period τ . The *reservoir* signal $s(t)$ travels in the feedback waveguide for total delay of τ and experiences attenuation by factor γ due to optical losses. The resultant signal $\gamma s(t)$ combines with $u(t+\tau)$ at the input to the *reservoir* MR, for the next τ step. Eq. 2.6-2.7 in Section 2.5.1 show how the *reservoir* MR (NL node) transforms $(u(t)+\gamma s(t-\tau))$ into $s(t)$.

The output layer of the DFRC accelerator is connected to the *reservoir* layer with a splitter, which splits a fractional power of the optical signal $s(t)$ travelling in the feedback waveguide and transfers it to the output layer. At the output layer, we employ another MR that acts as a filter to sample $s(t)$ at each θ . Total N such samples obtained during τ period represents the state of the *reservoir* for τ period. These N samples are converted by a photodiode (PD) into electrical domain, and are then digitized by the digitizer before being stored in the sample memory. These digitized samples in the sample memory are used for training the output weights that can be stored in the weight bank. During the testing/operating phase, the trained weights from the weight bank can be used to transform the sampled *reservoir* state to predict the output of the task.

2.4.2 Modelling the Nonlinear Response of the Reservoir MR

From [19], the richly nonlinear response (due to TPA) of an active MR's through-port transmission depends on the photon lifetime (τ_{ph}) of the MR cavity. For an MR, τ_{ph} depends on the MR's Q -factor. Therefore, we propose to control the nonlinearity of the *reservoir* MR (NL node), by controlling the Q -factor (hence, τ_p) of the MR. We can vary the Q -factor (hence, τ_{ph}) by doping the MR with a PN-junction and applying a reverse bias voltage across it to change the Q -factor (hence, τ_{ph}) [146]. We model the τ_{ph} dependent NL through-port response of the *reservoir* MR using Eq. 2.4-2.5.

$$s(t) = (u(t) + \gamma s(t-\tau))(1 - e^{(-\theta/\tau_{ph})}) + s(t-\tau), \quad \text{if } u(t) > s(t-\tau) \quad (2.4)$$

$$s(t) = (u(t) + \gamma s(t-\tau))(1 - e^{(-\theta/\tau_{ph})}) + s(t-\tau)e^{(-\theta/\tau_{ph})}, \quad \text{if } u(t) < s(t-\tau) \quad (2.5)$$

2.5 Evaluation

2.5.1 Evaluation Setup

We evaluate the performance of our proposed MR-based DFRC accelerator for three RC tasks that include two time series prediction tasks such as NARMA10 [76] and Santa Fe [4], and the third Nonlinear Channel Equalization task [77]. The timeseries prediction tasks have important application, both in engineering and medical care [83]. The computational abilities of the proposed DFRC accelerator are examined using the Normalized Root Mean Square Error (NRMSE) and Symbol Error Rate (SER). Our proposed MR-based DFRC accelerator (henceforth, identified as ‘Silicon MR’) is compared with the Mackey-Glass (MG) differential delay model based electronic DFRC accelerator [5] (henceforth, identified as ‘Electronic (MG)’) and the MZI-based all optical DFRC accelerator [177] (identified as ‘All Optical (MZI)’). In our setup, we use the binary masking technique proposed in [12], which uses maximum length sequences (MLS) to generate optimal mask pattern that is suitable across various tasks. To ensure ideal comparison with prior works, we employ the same binary masking technique across all considered DFRC accelerators. We analyzed the prediction errors and training time for ‘Electronic (MG)’, ‘All Optical (MZI)’ and ‘Silicon MR’ accelerators across the considered benchmark tasks. We also report and discuss the total power consumption for our considered photonic DFRC accelerators.

2.5.2 Error Metrics

Each considered benchmark task has a dataset, which consists of input signals and corresponding target output. Generally, the dataset is divided into two subsets called training and test sets. The training set is used to train the output weights of the DFR with the goal to predict the target output. The deviation in the target output from the predicted output is called error. The error reported for the DFRC accelerator on the test set gives the performance; lower the error better the DFRC accelerator performance. Below are the error metrics we have used in this Chapter.

1. Normalized Root Mean Square Error(NRMSE)

We use NRMSE for NARMA10 and Santa Fe timeseries tasks. NRMSE is defined by Eq. 2.6 [48].

$$NRMSE = \sqrt{\frac{\sum_{i=0}^N (y_i - \hat{y}_i)^2}{\sigma_{\hat{y}}^2}} \quad (2.6)$$

where \hat{y} is the predicted output, y_i is the target output, N is the total length of test set, $\sigma_{\hat{y}}^2$ is the variance of target output.

2. Symbol Error Rate (SER)

SER is used only for the Nonlinear Channel Equalization task, in which the DFR reproduces the input symbol in a noisy channel. The SER is evaluated

using Eq. 2.7 [77].

$$SER = \frac{\text{Total number of correctly reproduced symbols}}{\text{Total Number of symbols}} \quad (2.7)$$

2.5.3 Prediction Error Evaluation

1. NARMA10

The Nonlinear Autoregressive Moving Average of 10th order (NARMA10) is a time series whose current output depends on the past ten outputs. It was introduced in [177]. NARMA10 is a standard benchmark task for RC with DFRC accelerators [177, 48]. For NARMA10, the input $i(k)$ is drawn from a uniform distribution in interval $[0, 0.5]$ and the target output $y(k+1)$ is given by Eq. 2.8.

$$y(k+1) = 0.3y(k) + 0.005y(k) \sum_{i=0}^9 y(k-i) + 1.5i(k)i(k-9) + 0.1 \quad (2.8)$$

The input $i(k)$ is converted to $j(t)$ by sample and hold, then the masking signal $m(t)$ generated by MLS technique [12] is multiplied with $j(t)$ to get $u(t)$. We generate NARMA10 series dataset with 2000 samples; 1000 samples for training and next 1000 samples for testing, as done in [48]. We report the NRMSE on the testing set. The masking signal, training set and testing set are kept constant across all three considered DFRC accelerators. The NRMSE values for ‘Electronic (MG)’, ‘All Optical (MZI)’ and ‘Silicon MR’ accelerators are shown in Fig. 2.5. The value of NRMSE depends on the number of virtual nodes (N), and therefore, we do a sensitivity analysis to find the optimal value of N for each DFRC accelerator to get the least possible NRMSE. Owing to the space constraints we do not report all results obtained from the sensitivity analysis. In addition, for ‘Silicon MR’, photonic lifetime (τ_{ph}) (Eq. 2.4-2.5) also directly affects NRMSE. We find the minimum NRMSE for ‘Silicon MR’ at $N = 900$ and $\tau_{ph} = 50$ ps. Similarly, minimum NRMSE is achieved for ‘All Optical (MZI)’ and ‘Electronic (MG)’ at $N = 400$ and 900 , respectively. Moreover, ‘Silicon MR’ achieves 35% lower NRMSE compared to ‘All Optical (MZI)’, and it performs on par with ‘Electronic (MG)’, with ‘Silicon MR’ having very high training speed as discussed in Section 2.5.4. These results clearly corroborate the capabilities of ‘Silicon MR’ to perform complex RC tasks with good performance.

2. Santa Fe Time Series

The Santa Fe timeseries was introduced in [4], which has various datasets labelled A to F. Santa Fe dataset-A is a widely used for evaluating DFRC accelerators [46]. Santa Fe dataset-A records a far-infrared laser operating in chaotic state values. The goal is to predict the laser behavior one-time step ahead. We have used an extended version of Santa Fe dataset-A with 6000 samples from [141]. The dataset is split into 4000 training samples and 2000 testing samples. We obtain the lowest NRMSE for ‘Silicon MR’ at $N=40$ and $\tau_{ph}=50$ ps.

As illustrated in Fig. 2.5, ‘Silicon MR’ performs extremely well on Santa Fe compared to ‘All Optical (MZI)’ with 98.7% lower NRMSE. Even though ‘Electronic (MG)’ achieves a little better NRMSE, it requires $N=400$, which is $10\times$ the N required by ‘Silicon MR’.

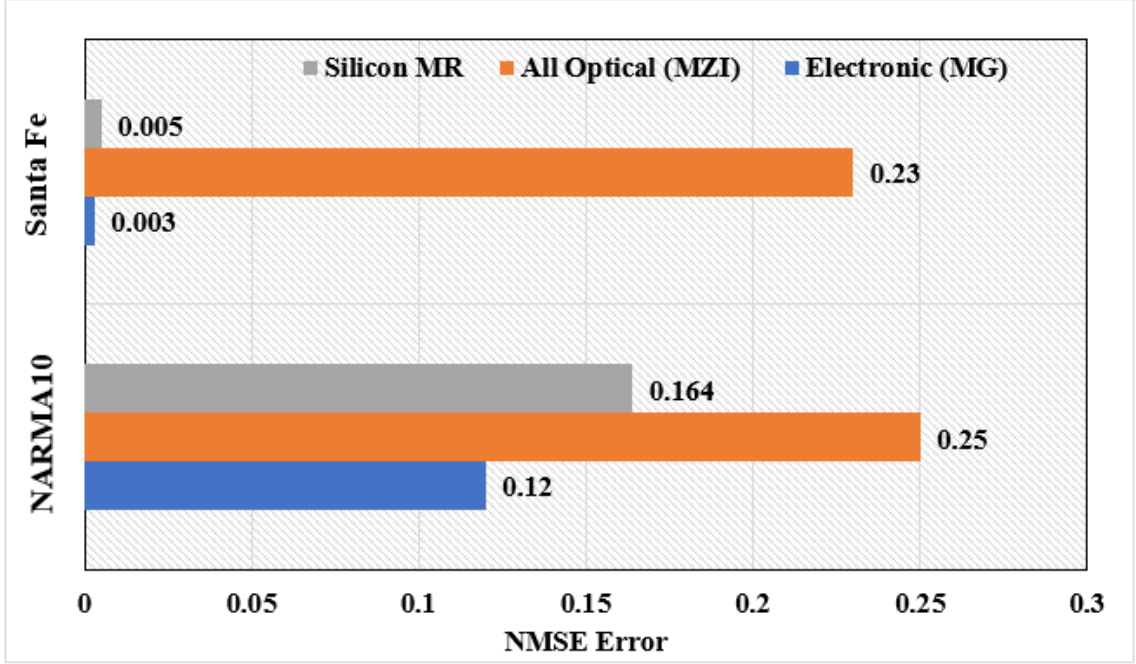


Figure 2.5: NRMSE values for ‘Silicon MR’, ‘All Optical (MZI)’ and ‘Electronic (MG)’ for NARMA10 and Santa Fe timeseries tasks.

3. Nonlinear Channel Equalization Task

In this task, the goal of the DFRC accelerator is to reproduce the distorted symbol due to noise in a wireless communication channel. It is a common task which investigates the nonlinearity of DFRC accelerators. The input-output relation is defined by Eq. 2.9-2.10 [77].

$$\begin{aligned}
 q(n) = & 0.08d(n+2) - 0.12d(n+1) + d(n) + 0.18d(n-1) \\
 & - 0.1d(n-2) + 0.09d(n-3) - 0.05d(n-4) \\
 & + 0.04d(n-5) + 0.03d(n-6) + 0.01d(n-7) \quad (2.9)
 \end{aligned}$$

$$x(n) = q(n) + 0.036q^2(n) - 0.011q^3(n) + v(n) \quad (2.10)$$

where $d(n)$, is an independent, identically distributed, four level $\{-3, -1, 1, 3\}$ sequence, $v(n)$ is a pseudo-random Gaussian sequence with zero mean and variance determined by the desired output signal-to-noise ratio (SNR). We vary SNR from 12dB–32dB with a step size of 4dB. We generate 9000 symbols for

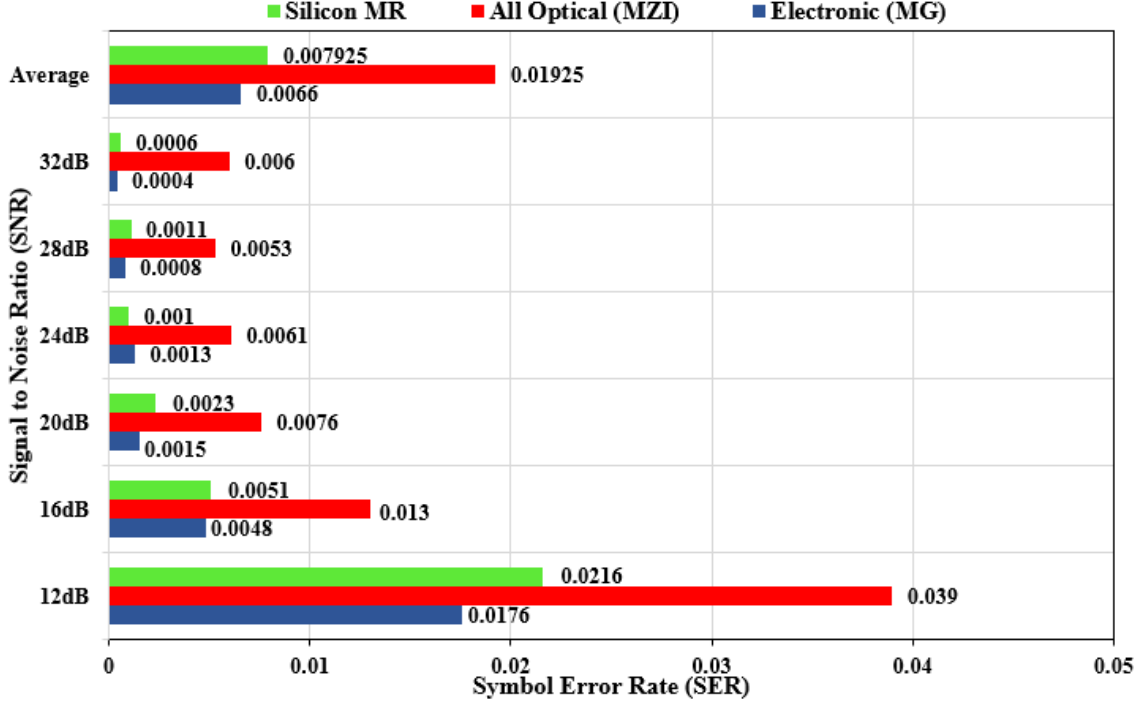


Figure 2.6: SER values of ‘Silicon MR’, ‘All Optical (MZI)’ and ‘Electronic (MG)’ for Nonlinear Channel Equalization task, with SNR ranging from 12dB-32dBs.

a dataset, of which 6000 are for training and 3000 are for testing. To evaluate DFRC accelerators for Nonlinear Channel Equalization, we use SER metric. The best SER for ‘Silicon MR’ is found at $N=30$ and $\tau_{ph} = 50\text{ps}$. The SERs of ‘Silicon MR’, ‘Electronic (MG)’ and ‘All Optical (MZI)’ are presented in Fig. 2.6. Across almost all SNR values, the best SER is achieved for ‘Electronic (MG)’, closely followed by ‘Silicon MR’ and highest SER for ‘All Optical (MZI)’. Nevertheless, ‘Silicon MR’ reaches 23% lower SER than ‘Electronic (MG)’ for 24dB SNR. On average, ‘Silicon MR’ outperforms ‘All Optical (MZI)’, having 58.8% lower SER.

2.5.4 Training Time

Fig. 2.7 gives the time consumed by each DFRC accelerator to complete the training of the output weights. The training time includes time required to observe the DFR states for each input and time required to train the output weights using linear regression. The time involved in generating the DFR states depends on the total delay τ associated with the feedback loop. ‘All Optical (MZI)’ and ‘Silicon MR’ clearly have the advantage of shorter training time due to their shorter τ values of $7.56\mu\text{s}$ and 45ns , respectively, compared to $\tau = 10\text{ms}$ for ‘Electronic (MG)’. Furthermore, the integrated dynamics of ‘Silicon MR’ yield τ_{ph} and θ to operate at picosecond

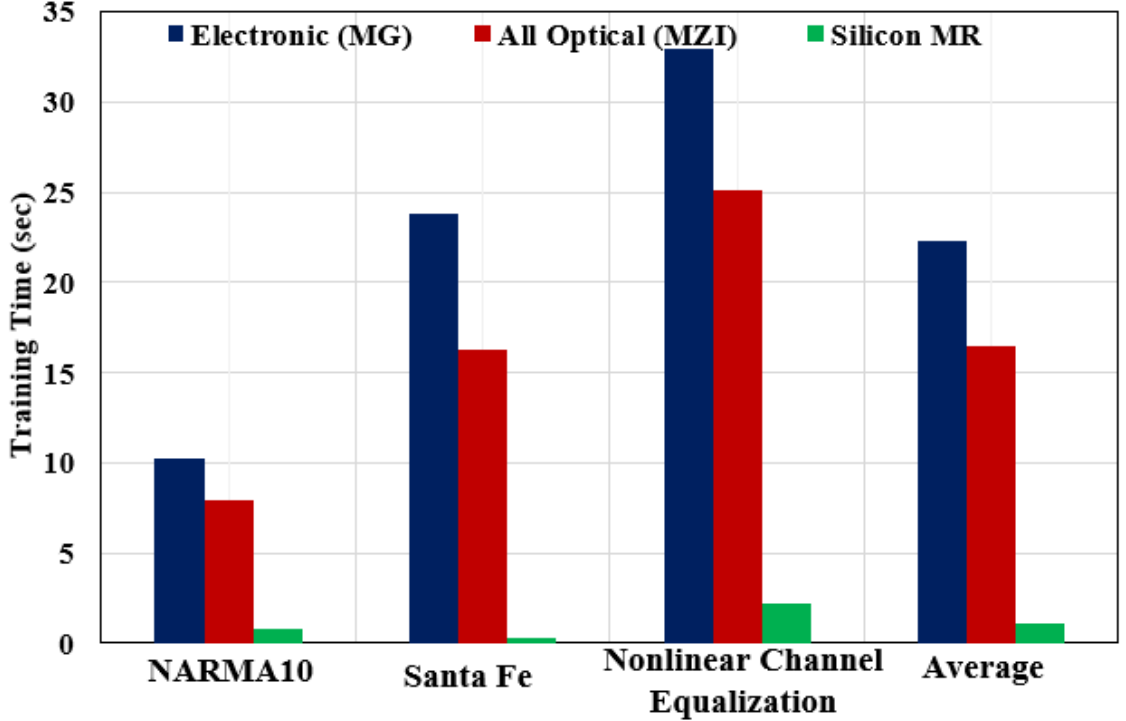


Figure 2.7: Training time of ‘Silicon MR’, ‘All Optical (MZI)’ and ‘Electronic (MG)’ for tasks NARMA10, Santa Fe and Nonlinear Channel Equalization.

scale, resulting in a significant speedup of training for ‘Silicon MR’ by $98\times$ and $93\times$ on average compared to ‘All Optical (MZI)’ and ‘Electronic (MG)’ respectively.

2.5.5 Discussion on Power Consumption

We report and discuss power consumption of the photonic DFRC accelerators ‘Silicon MR’ and ‘All Optical (MZI)’. In contrast, as the ‘Electronic (MG)’ implementation in [5] used all computer-controlled off-the-shelf electronic components, we are not able to report the exact power numbers for it. The necessity of ADC and DAC in the *reservoir* layer of ‘Electronic (MG)’ hinders its throughput scaling and increases the power consumption compared to photonic DFRC accelerators. Since the operations of the input and output layers are identical in both photonic DFRC accelerators, major power variations can be noticed in the *reservoir* layer. For the photonic *reservoir* in ‘Silicon MR’ and ‘All Optical (MZI)’, we calculate the required laser power P_{Laser} using Eq. 2.11.

$$P_{Laser} = IL^{dB} + CouplingLoss + SplitterLoss + DynamicRange + S \quad (2.11)$$

where IL^{dB} is the total insertion loss, dynamic range is optical power range required to implement masking of optical signal, and S is the sensitivity of the photodetector. We considered the values listed in Table 2.1, and evaluated the total power consumption

Table 2.1: Various Loss And Power Parameters

Parameter	‘Silicon MR’	‘All Optical (MZI)’
Laser wall-plug efficiency	10% [19]	10% [19]
PD Sensitivity at 10Gb/s	-5.8dBm [177]	-5.8dBm [177]
IL^{dB}	8.25dB [113]	7.4 dB [48]
Splitter Loss	0.5dB [66]	NA
Coupling Loss	2dB [66]	3.3 dB [113]
Free Spectral Range (FSR)	20 nm [141]	NA
Dynamic Range	6dB	20dB [48]
ZHL-32A amplifier	NA	10 dBm [48]
Feedback Photodiode (TTI TIA525)	NA	1.2mW [48]
Optical Attenuator (Agilent 81571A)	NA	33dBm [48]
MR Modulator	15fJ/bit [177]	NA
MR filter	0.705pJ/bit [177]	NA
MZI modulator	NA	100mW [48]

(laser + dynamic for all layers) in ‘Silicon MR’ to be 126.48mW and in ‘All Optical (MZI)’ to be 549.54mW. Due to the high optical resolution of the *reservoir* MR used in ‘Silicon MR’, compared to the MZI modulator used in ‘All Optical (MZI)’, ‘Silicon MR’ requires lower dynamic range (Table 2.1) to implement the masking of the input optical signal, thereby, requiring total overall power.

2.6 Summary

In this Chapter, we presented a silicon MR based chip-scale accelerator for delayed feedback *reservoir* computing (DFRC). Our DFRC accelerator leverages the rich non-linearity of the active MR to realize the nonlinear node in the *reservoir* layer of the accelerator. Moreover, it uses a photonic waveguide as the feedback delay loop to enable fully on-chip integration of the *reservoir* layer. Evaluation with benchmark tasks shows that our MR-based DFRC accelerator achieves 35% and 98.7% lower NRMSE, up to 58.8% less average SER, and up to 93x faster training time compared to a photonic DFRC accelerator from prior work. Thus, our MR-based DFRC accelerator represents an attractive solution for realizing scalable *reservoir* computing for integrated applications.

Chapter 3 Rule-Based Self-Adaptation in Photonic NoCs for Loss-Aware Co-Management of Laser Power and Performance

3.1 Introduction

To support the increasing demand for on-chip data communication in modern multi-core processors, the use of electrical networks-on-chip (ENoCs) has become a norm. However, the performance of the state-of-the-art ENoCs is projected to scale poorly for the emerging data-centric applications (e.g., internet-of-things (IoT) related applications), primarily due to the energy-constrained bandwidth of ENoCs. To this end, with the recent advancements in silicon photonics, photonic networks-on-chip (PNoCs) are being considered as potential replacements for ENoCs. This is because PNoCs can provide several advantages over ENoCs, such as distance-independent higher datarates and lower dynamic energy consumption. However, the state-of-the-art PNoC architectures (e.g., [124], [80]) require a non-trivial amount of optical power from their laser source, mainly because of the high insertion loss of photonic devices in their constituent photonic links [66]. The high laser power overheads can offset the high aggregated datarate and energy-efficiency advantages of PNoCs. Therefore, it is imperative to innovate new techniques that can reduce the optical power consumption in future PNoC architectures.

Several techniques have been proposed in prior works (e.g., [212, 121, 33, 32, 84, 194, 172, 182, 23]) that aim to reduce the laser power consumption in PNoCs. Some of these techniques dynamically adjust the optical power extracted from the off-chip laser sources, in response to either the temporal and spatial variations in the network traffic (e.g., [212, 121, 33, 32, 84]) or the change in the insertion loss for every photonic data packet transfer (e.g., [194, 172]). In addition, recent works [182] and [23] take a holistic approach and use machine learning predictors for leveraging the variations in both the network traffic and insertion loss, to achieve greater savings in laser power consumption. However, all these techniques can incur dauntingly high overheads for dynamic monitoring of the network traffic (e.g., in [212, 121, 33, 32, 84]), integration of costly on-chip optical amplifiers (e.g., in [172]), runtime execution of NP-hard optimization heuristics (e.g., in [194]), or runtime inference of the machine learning models (e.g., in [182], [23]). Moreover, these techniques do not consider the effects of bit-error rate (BER) penalty due to various sources of errors (e.g., cross-talk) on the laser power utilization and performance of photonic links and PNoCs. As a result, these techniques are not able to achieve the required practical balance between the reduction in laser power and achieved performance in PNoCs. To achieve such power-performance balance in PNoCs, recent works [98] and [158] employ data approximation techniques to opportunistically trade the communication reliability for reduced laser power and/or improved performance in PNoCs. However, to gain substantial benefits, these techniques require the accuracy or reliability goals of target applications to be relaxed, which can be achieved only for a select few inherently error-tolerant applications. This constraint limits the applicability of such techniques.

In contrast to these dynamic techniques from prior work, we advocate for a hybrid (static + dynamic) solution in this Chapter as part of our proposed *PROTEUS* framework. Instead of dynamically tuning the optical power extracted from the laser source, our *PROTEUS* framework statically minimizes the required optical power extraction from the laser source at the design-time, by optimizing two key photonic link configuration parameters to minimize the BER power penalty in PNoCs without reducing the reliability of communication. Then, at the runtime, *PROTEUS* dynamically adapts the photonic link configuration in response to the changing insertion loss for every photonic packet transfer, to achieve and maintain the balance between the reduction in laser power and achieved performance. For dynamic adaptation, *PROTEUS* relies on simple rules that are derived from an offline search heuristic. *PROTEUS* stores these rules in lookup tables to enable their easy reference during the runtime of PNoCs. Our novel contributions in this Chapter are summarized below:

- We present a design-time technique that minimizes the crosstalk related BER power penalty in PNoCs by optimizing two key photonic link configuration parameters, to ultimately reduce the requirement of laser power in PNoCs;
- We present light-weight techniques for implementing self-adaptation of photonic link configuration, and provide detailed overhead analysis of these techniques;
- We integrate these design-time optimization and runtime self-adaptation techniques into a holistic framework called *PROTEUS*, to achieve a loss-aware balance between the laser power consumption and performance of PNoCs;
- We evaluate *PROTEUS* by implementing it on a well-known PNoC architecture and compare it with other laser power management techniques from prior works [33] and [172].

3.2 Fundamentals of Photonic NoCs (PNoCs)

3.2.1 Physical-Layer Architecture and Operation of PNoCs

In this subsection, we explain the physical-layer design and operation of PNoC architectures. We use the crossbar-based PNoC from [35] as an example PNoC architecture in this Chapter, the physical-layer layout of which is illustrated in Fig. 3.1. The PNoC in Fig. 3.1 consists of serpentine links as its building blocks. Every such link in the PNoC consists of one or more photonic waveguides spanning the PNoC chip, depending on the specific variant of the physical-layer architecture [25]. In this Chapter, we consider one photonic waveguide per link. Every such single-waveguide photonic link in the PNoC connects multiple gateway interfaces (GIs) with one another. A GI connects to multiple parallelly laid-out photonic links, and interfaces a cluster of processing cores (e.g., a cluster of four cores in Fig. 3.1) with the links. Typically, out of all the GIs that are connected to a single link, some GIs can write photonic data into the link and the others can read photonic data from the link, to

for data signals, to enable data communication with one or more other GIs. At a sender GI of the PNoC, every incoming data packet from the source processing core is converted into multiple parallel electrical data signals (a signal is defined here as a sequence of ‘1’s and ‘0’s), which are then modulated onto the DWDM carriers using a bank of modulator MRs (not shown in Fig. 3.1) to convert them into parallel photonic data signals. These DWDM data signals constitute a photonic data packet that traverses a single-waveguide link to a receiver GI. At the receiver GI, a set of MR filters drops the constituent photonic signals of the photonic data packet onto the adjacent photodetectors, to regenerate the electrical data signals, and consequently, the electrical data packet. This regenerated electrical data packet is then passed on to the destination processing core. Thus, in a PNoC, every data packet is transferred as multiple DWDM data signals.

The transfer of every photonic data packet as multiple DWDM data signals (referred to as N_λ) in the PNoC enables wrapping of the data packet (packet size is referred to as PS) into a short timeframe. This packet timeframe (i.e., $|(PS/N_\lambda)| \times (1/BR)$, where BR is signal bitrate) is referred to as frame delay in Fig. 3.1. This frame delay, when added to the waveguide propagation delay (Fig. 3.1), constitutes the latency of transferring the packet between the sender and receiver GIs. It can be reasoned that this transfer latency for a fixed size of the data packet can be reduced by decreasing the packet frame delay (i.e., $|(PS/N_\lambda)| \times (1/BR)$), which in turn can be achieved in three different ways: (i) by increasing N_λ in the waveguide, (ii) by increasing the bitrate (i.e., BR) of each data signal, and (iii) by increasing both N_λ and BR . Each of these three ways can enable wrapping of the data packet into a shorter timeframe, to reduce the packet frame delay, and hence, the packet transfer latency. However, increasing N_λ and/or BR requires judicious consideration of the inherent tradeoffs among the achievable performance, reliability, and required optical power in the PNoC. Failing to do so can lead to significantly harmed optical power efficiency or nonviable operation of the photonic links and PNoC, as discussed next.

3.2.2 Power-Reliability-Performance Tradeoffs in PNoCs

Designing a photonic link of a PNoC is subject to inherent tradeoffs among the achievable performance (aggregated data rate ($N_\lambda \times BR$), and hence, frame delay ($|(PS/N_\lambda)| \times (1/BR)$, where PS is packet size), required optical power, and reliability [19]. Optimizing these design tradeoffs often involves finding the sweet spot that balances the link’s aggregated data rate and power-reliability behavior [19]. The tenacity of this balance depends on how efficiently the provisioned optical power from the off-chip laser source is utilized. The utilization of the provisioned laser power in the link is governed by four different factors, which are formulated in Eq. 3.1.

$$P_{Max} \geq P_{Laser} \geq IL^{dB} + PP^{dB}\{N_\lambda, BR, Q\} + 10\log_{10}^{(N_\lambda)} + S\{BR\} \quad (3.1)$$

Here, P_{Laser} is the provisioned optical power (in dBm) in the link from the power-waveguide splitter (Fig. 3.1), the utilization of which depends on the following four factors, as evident from Eq. 3.1: (i) total insertion loss IL^{dB} in dB faced by a single

photonic signal in the link, which includes the total propagation and bending loss in the link's waveguide and the total insertion loss of the MR modulators, MR filters, splitters, and couplers; (ii) total bit-error rate (BER) power penalty PP^{dB} , which is defined as the required increase in the provisioned optical power of a photonic signal to compensate for the reduced bit-error rate (BER) due to various signal degradation phenomena, including intermodulation crosstalk and inter-signal crosstalk at filter MRs [17]; (iii) number of DWDM data signals N_λ per waveguide; and (iv) the photodetector sensitivity S which is a function of BR , which gives the minimum required power of a photonic data signal at the photodetector for the error-free detection of the signal. In addition, the peak value of P_{Laser} in a link should be less than P_{Max} (Eq. 3.1), where P_{Max} gives the optical nonlinearity limited maximum allowable optical power in the waveguide (typically, $P_{Max} = 20\text{dBm}$ [175], [19]). Thus, P_{Laser} in the link should be not only greater than or equal to the sum of the optical power requirements of all the above four factors, but also less than or equal to P_{Max} .

From [19], S depends on the BR of the photonic signal. Similarly, from [18] and [17], the total power penalty PP^{dB} of a link, as well as the insertion loss values for the modulator and filter MRs (which are part of the total IL^{dB} value for the link), also depend on BR . In addition, PP^{dB} of a link also depends on various link configuration parameters, such as quality factor (Q) of the MRs, free spectral range (FSR), and wavelength spacing between the adjacent photonic signals in the link [66]. The parameters wavelength spacing and FSR have limited design flexibility due to the limitations imposed by the utilized devices and fabrication technology [19]. For instance, commonly used comb laser sources typically produce output wavelengths with precisely fixed spacings [136], and require additional area-consuming interleavers (e.g., [134]) to provide limited flexibility for tuning their output wavelength spacings. Along the same lines, the state-of-the-art CMOS-compatible MR fabrication technology limits the maximum achievable FSR to 20nm (e.g., [141]). Because of these reasons, for the system-level design of PNoCs, the values of parameters FSR and wavelength spacing can be assumed to be fixed, and consequently, PP^{dB} can be optimized as the function of BR and Q of MRs (see Section 3.2.3). As a result, the required P_{Laser} and its utilization in the photonic link ultimately depends on the link configuration parameters N_λ , Q , and BR . Thus, for the given value of IL^{dB} in the link, only a finite set of unique values of the (N_λ, BR, Q) triplet can satisfy the condition for P_{Laser} given in Eq. 3.1. From [174] and [175], out of all such values of triplet (N_λ, BR, Q) , only one triplet value can optimally balance the inherent tradeoffs among the aggregated data rate ($N_\lambda \times BR$), frame delay (packet size / ($N_\lambda \times BR$)), and optical power efficiency ($P_{Laser} / (N_\lambda \times BR)$). Thus, any injudicious attempt to increase N_λ for improving the packet frame delay can lead to an increased P_{Laser} value, which in turn can result not only in a decreased optical power efficiency ($P_{Laser} / (N_\lambda \times BR)$), but also in a nonviable P_{Laser} value that is greater than P_{Max} .

3.2.3 Modeling of PP^{dB} and IL^{dB} as Functions of BR and Q

From [13], Eq. 3.2 below gives the formula for PP^{dB} (from Eq. (3.1)) for a photonic signal as the sum of the modulator crosstalk penalty (PP_{Xtalk}^{Mod}), filter crosstalk penalty

(PP_{Xtalk}^{Fil}), and power penalty due to the finite Extinction ratio (ER) of modulation (i.e., the first term in Eq. 3.2). From [18], PP_{Xtalk}^{Mod} for a signal does not depend on its BR , and for a moderate wavelength spacing of greater than 0.3nm (as assumed for this work), it can be limited below 1dB. Therefore, we take the fixed 1dB value of PP_{Xtalk}^{Mod} in this Chapter. On the other hand, PP_{Xtalk}^{Fil} at a filter MR can be evaluated using Eq. 3.3 and Eq. 3.4 given below [18].

$$PP^{dB} = -10\log_{10}\left(\frac{r-1}{r+1}\right) + PP_{Xtalk}^{Mod} + PP_{Xtalk}^{Fil} \quad (3.2)$$

$$PP_{Xtalk}^{Mod} \approx -10\log_{10}\left(1 - 2\sum_{i=1}^{N_\lambda} \sqrt{\gamma_i}\right) \quad (3.3)$$

$$\gamma_i = \frac{1}{1+\beta^2} - \frac{1}{2\pi v} \text{Re}\left(\frac{1 - \exp(-2\pi v(1-j\beta))}{(1-j\beta)^2}\right) \quad (3.4)$$

Here, r extinction power ratio, γ_i is the crosstalk power ratio at the filter MR from the i_{th} signal of total N_λ signals, $v=f_0/(2Qr_b)$, $\beta=2Qf_\delta/f_0$, with $Q = \text{MR } Q$, $r_b = BR$ of the i_{th} signal, f_0 is resonance frequency of the MR filter, and f_δ denotes the frequency detuning between the i_{th} signal and f_0 . Fig. 3.2 gives the modeled PP_{Xtalk}^{Fil} values as a function of BR and Q . From the figure, for a given value of BR , only a unique value of Q (as indicated by the optimal curve) can minimize PP_{Xtalk}^{Fil} .

In addition, from [18], the insertion losses of MR modulators and filters can be modeled to depend on their Q using the Lorentzian shaped transfer function of MRs. The inclusion of these Q -dependent insertion loss values of MR modulators and filters in the total IL^{dB} value for the link makes IL^{dB} to depend on the MRs' Q as well. *Thus, only a unique combination of Q and BR can minimize both PP^{dB} and IL^{dB} for a photonic link.*

3.2.4 Variation in IL^{dB} for Every Photonic Packet Transfer

In a PNoC, different photonic data packets face different values of the insertion loss IL^{dB} . This is because different photonic packets traverse different distances between their sender and receiver GIs. For example, in Fig. 3.1, a source processing core is highlighted as SR. The photonic packets from SR traverse paths P1 and P2, respectively, to the destination processing cores D1 and D2. Based on the physical layout of the PNoC shown in Fig. 3.1 on a 2cm×2cm photonic chip for the 22nm technology node, the lengths of paths P1 and P2 are 1.74cm and 2.61cm respectively. Moreover, path P2 also has a waveguide bend. Therefore, based on the various loss model values from Table 1, paths P1 and P2 incur waveguide propagation loss of 0.94dB and 1.41dB respectively. This in turn makes the insertion loss IL^{dB} value (that includes the waveguide propagation loss in addition to some other loss parameters [66]) to change for each data packet transfer in the PNoC. This observation opens new opportunities for dynamically changing for every packet transfer either the P_{Laser} value or the utilization (in terms of PP^{dB} and/or N_λ) of the fixed design-time P_{Laser} value.

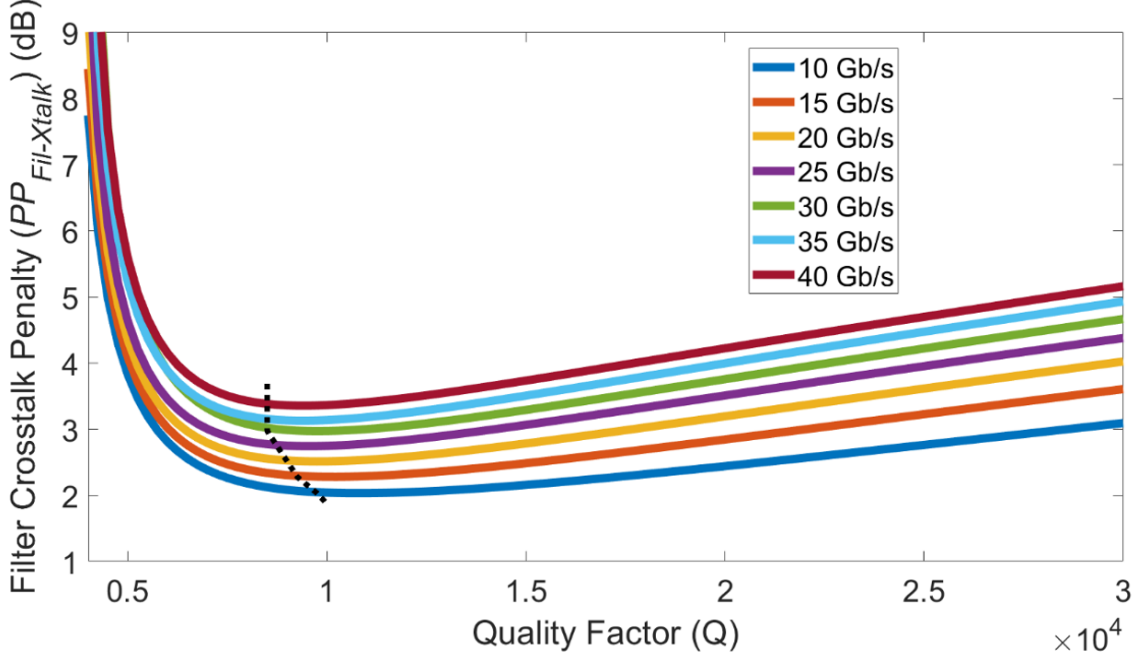


Figure 3.2: Filter Crosstalk Penalty (PP_{Xtalk}^{Fil}) as a function of quality factor (Q) for various values of signal Bitrate (BR). Evaluation is done using Eq. 3.2 and Eq. 3.3 for 0.37nm wavelength spacing and $N_\lambda=55$ at 1550nm operating wavelength

3.3 Related Work And Motivation

Because of the high insertion loss and power penalty in the constituent photonic links [18, 175, 66], the state-of-the-art PNoC architectures require a non-trivial amount of optical power from their laser source. The high optical power overheads from the laser source can offset the high aggregated datarate, low packet frame delay, and optical power efficiency advantages of PNoCs. Therefore, it is imperative to innovate new techniques that can reduce the optical power consumption in future PNoC architectures. Several prior works have addressed this problem, as discussed next.

3.3.1 Prior Works on Laser Power Management

Several techniques have been proposed in prior works (e.g., [212, 121, 33, 32, 84, 194, 172, 182, 23]), that aim to reduce the optical power consumption, and hence, the power consumption of laser sources in PNoCs. To achieve the power savings, a few of these techniques (e.g., [212, 121, 33, 32, 84]) leverage the temporal and spatial variations in network traffic to opportunistically adjust the P_{Laser} value (i.e., optical power extracted from laser sources) by tuning or distributing the available N_λ in the network. These methods tend to notably reduce the power in laser sources during low network load conditions. However, if the losses encountered by optical signals

in the network between the sender and receiver GIs are high, these methods would still require excessive optical power from laser sources to compensate for the high losses, even under low network load conditions. In contrast, a few other techniques focus (e.g., [194], [172]) on leveraging the inherent change in IL^{dB} per packet transfer to tune the P_{Laser} value (output optical power from laser sources). The amount of optical power savings achieved by these methods depends on how often the P_{Laser} value can be tuned in response to the changing IL^{dB} . In addition, recent works [182] and [23] take a holistic approach and focus on both adapting N_λ and leveraging the change in IL^{dB} using machine learning predictors, to achieve greater savings in optical power consumption.

In addition, recent works [98] and [158] employ data approximation techniques to opportunistically trade the communication reliability for reduced laser power and/or improved performance in PNoCs. However, to gain substantial benefits, these techniques require the accuracy or reliability goals of target applications to be relaxed, which can be achieved only for a select few inherently error-tolerant applications. This constraint limits the applicability of such techniques.

3.3.2 Motivation for Rule-Based Self-Adaptation in PNoCs

Techniques from prior work that look to dynamically adapt N_λ in response to the changing network traffic conditions need to incorporate extra mechanisms with PNoCs to (i) monitor the network traffic conditions at runtime, (ii) distribute the available N_λ in the network, and (iii) communicate the tuning decisions to the off-chip laser sources. The overheads of such extra mechanisms can offset the achieved optical power benefits. Along the same lines, among the techniques that look to leverage the change in IL^{dB} , [172] requires an integration of costly on-chip optical amplifiers, whereas [194] requires runtime execution of optimization heuristics. Moreover, the machine learning based self-adaptation techniques from [182] and [23] can also incur high overheads of runtime inference of the machine learning models. In addition, all these techniques do not consider the power penalty (PP^{dB}) as an important factor that can affect the utilization of P_{Laser} in PNoCs in terms of supported N_λ . As a result, these techniques often render infeasibly high N_λ values, failing to obtain the practical balance between the optical power efficiency and packet frame delay (packet transfer latency).

In contrast to these dynamic techniques from prior work, we advocate for a hybrid (static + dynamic) solution as part of our proposed *PROTEUS* framework that can achieve and maintain a balance between the optical power efficiency and performance of PNoCs. The details of our proposed *PROTEUS* framework are discussed in the next section.

3.4 Proposed PROTEUS Framework

3.4.1 Overview

Our proposed *PROTEUS* framework enables rule-based self-adaptation in PNoCs for dynamic management of P_{Laser} and performance (in terms of packet transfer latency). *PROTEUS* includes two steps. In the first design-time step (Section 3.5.2), *PROTEUS* performs a search heuristic based optimization to find the optimal combination of Q and BR that minimizes the PP^{dB} and IL^{dB} values for the link. This step allows *PROTEUS* to statically reduce the P_{Laser} value at the design time, compared to the techniques from prior works [172] and [33], and balance the optical power efficiency of the PNoC with its packet transfer latency. Then, during the second runtime step (Section 3.5.3), *PROTEUS* readjusts the BR , and Q duplet in response to the changing IL^{dB} for every packet transfer, (i) to ensure that the provisioned P_{Laser} is always utilized as fully as possible, and (ii) to maintain the balance between the achieved optical power efficiency and packet frame delay (packet transfer latency) for every packet transfer. To enable dynamic readjustments (adaptation) of Q , *PROTEUS* incorporates the MR modulator/filter design with adaptable Q from [146], after enhancing it for a faster response. Similarly, to enable adaptation in BR , *PROTEUS* allows a light-weight reconfiguration of the serialization and deserialization modules in each GI to enable the scaling of photonic clock rate (that directly corresponds to signal BR) between the baseline value of 5GHz and four discrete up-scaled values (10GHz, 15GHz, 20GHz, and 25GHz). An exhaustive search-based analysis is performed offline, to find the best combinations of Q and BR for all possible IL^{dB} values in the PNoC. From this offline analysis, simple rules are derived about what should be the change in the control parameters (e.g., reconfiguration parameters that control the dynamic clock rate scaling) to adapt BR and Q combination for each transferred packet, as IL^{dB} changes for each packet transfer as discussed in Section 3.2.4. These rules (i.e., new control parameter values) are stored in a lookup table at every GI of the PNoC, which *PROTEUS* refers to at runtime before each packet transfer to enable adaptation of Q and BR .

3.4.2 Search Heuristic Based Design-Time Optimization

In a PNoC, IL^{dB} varies for different sender-receiver pairs, as illustrated in Fig. 3.1 (Section 3.2.4). We model all unique IL^{dB} values that a photonic packet can experience across all possible sender-receiver combinations. For our PNoC in Fig. 3.1, the best-case IL^{dB} is 0.47dB and the worst-case IL^{dB} is 10dB. Note that we consider only the waveguide propagation loss as IL^{dB} for our analysis presented in this section. Prior works [172] (henceforth identified as OPA) and [33] (identified as ABM), with which we compare our *PROTEUS* framework, do not consider the impact of PP^{dB} on P_{Laser} utilization, as inferred from the fact that the assumed Q or BR values are not reported in [172] and [33]. As a result, OPA and ABM assume invariably high value of $N_\lambda = 64$ that leads to P_{Laser} to be greater than $P_{Max} = 20\text{dBm}$ [197], for commonly used fixed values of $Q = 7000$ [19] and $BR = 10 \text{ Gb/s}$ [19][182]. Therefore, to make the implementations of OPA and ABM techniques viable, first, we identify

the viable value of N_λ (using Eq. (3.1)) for the worst-case IL^{dB} of 10dB. For that, we consider $Q=7000$, $BR=10\text{Gb/s}$, $S = 20\text{dBm}$ [19], $P_{Laser}=P_{Max}=20\text{dBm}$, and PP^{dB} as evaluated from Eq. (3.2)-(3.4). We found the maximum supported N_λ to be 55, and we consider this as the design value for OPA and ABM. As our *PROTEUS* framework aims to achieve loss-aware power savings, we consider another loss-aware technique, i.e., OPA, as the baseline comparison in this section. Fig. 3.3 gives the packet frame delay and optical power efficiency (triangle shaped points) for different IL^{dB} values (shown in different colors) for OPA. As evident, OPA reduces P_{Laser} as IL^{dB} decreases. As a result, the optical power efficiency values for OPA also reduce as IL^{dB} decreases. However, as $N_\lambda=55$ and $BR=10\text{Gb/s}$ are fixed for all IL^{dB} cases for OPA, all IL^{dB} cases achieve the same packet frame delay (Fig. 3.3). From these results for OPA, the goal of *PROTEUS* framework becomes to statically reduce the required P_{Laser} to a value below 20dBm that can support the unchanged aggregated datarate ($N_\lambda \times BR=55 \times 10\text{Gb/s}=550\text{Gb/s}$) for all possible IL^{dB} cases. Intuitively, if a P_{Laser} value that is less than 20dBm can support $N_\lambda=55$ for the worst-case IL^{dB} of 10dB, then that P_{Laser} value can support $N_\lambda=55$ for all other IL^{dB} values lower than 10dB as well. To find such P_{Laser} value, *PROTEUS* aims to reduce PP^{dB} for the worst-case $IL^{dB} = 10\text{dB}$, by optimizing Q for the given $BR = 10\text{Gb/s}$ (unchanged compared to OPA), using a search heuristic. The search heuristic takes 28 different Q values (i.e., from 5000 to 12000 with step increment of 250) and finds $Q = 9750$ to provide minimal PP^{dB} for $BR = 10\text{Gbps}$, which corroborates with Fig. 3.2 where the optimum curve for $BR = 10\text{Gbps}$ falls at the same value of $Q = 9750$. Thus, at $Q = 9750$ we have the least PP^{dB} , which gives us the opportunity to statically reduce P_{Laser} .

3.4.3 Impact of Varying Q and BR

From Section 3.4.2, intuitively any IL^{dB} that is less than the worst-case value of 10dB should require less than $P_{Laser}=16\text{dBm}$. But *PROTEUS* keeps P_{Laser} to be fixed at 16dBm for each packet transfer, irrespective of IL^{dB} . This provides an opportunity to increase BR for smaller IL^{dB} values, by allowing the accommodation of a larger PP^{dB} value to fully utilize the provisioned P_{Laser} of 16dBm. For fully utilizing the provisioned P_{Laser} for different IL^{dB} values, *PROTEUS* adaptively varies Q and BR for different IL^{dB} value (i.e., for each different packet). For that, *PROTEUS* uses the offline search heuristic to find the optimal values of Q , BR that provides the minimum positive value of $e = (P_{Laser} - IL^{dB} - PP^{dB} - 10\log(N_\lambda) - S)$ (derived from Eq. (3.1)), as the minimum value of e means that P_{Laser} is fully utilized for that BR and Q combination. Such optimal BR and Q values are found for each possible IL^{dB} value in the PNoC. As inputs to the search heuristic, we use the same values of Q as used in Section 3.4.2, whereas we limit BR to only four discrete values of 10Gbps, 15Gbps, 20Gbps, and 25Gbps to enable a viable BR adaptation control mechanism as discussed in Section 3.5.1. From Fig. 3.3, as the IL^{dB} values reduce from 10dB, the optimal Q and BR values for *PROTEUS* change, yielding increasingly better (lower) frame delay and optical power efficiency values. To understand the reason behind that, consider Fig. 3.3 that plots the breakdown of P_{Laser} utilization and aggregated

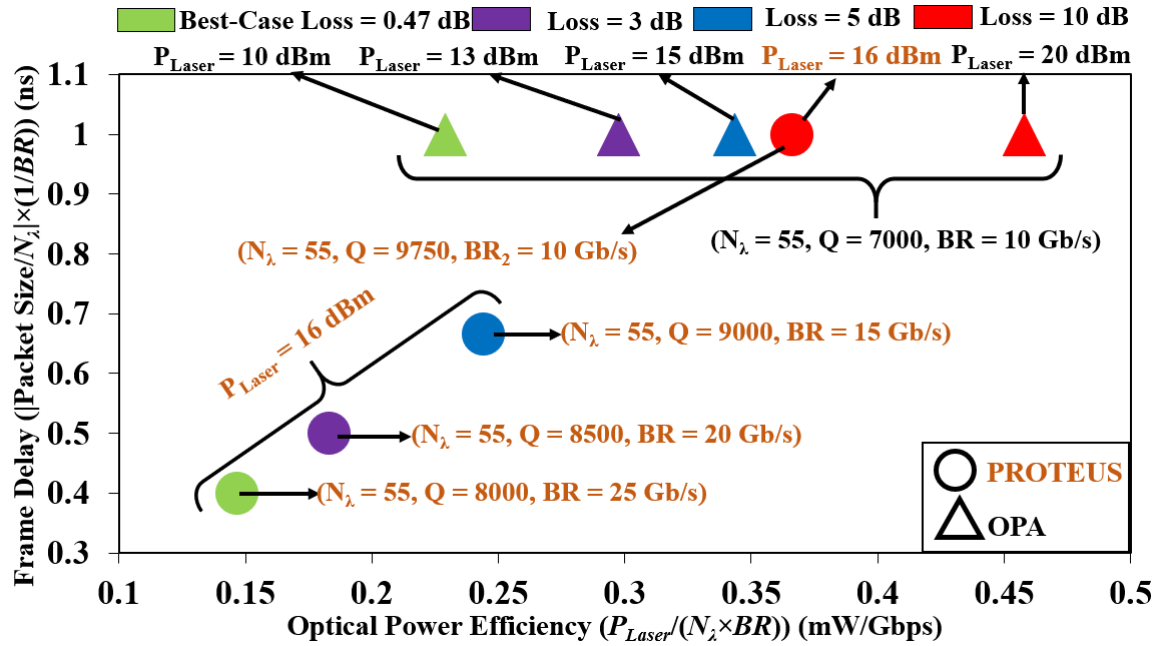
datarate values for OPA and *PROTEUS* for various insertion loss (IL^{dB}) values. In Fig. 3.3, P_{Laser} decreases for OPA as the insertion loss (IL^{dB}) decreases. In contrast, for *PROTEUS*, P_{Laser} remains constant for all insertion loss values. However, the detector sensitivity increases as the insertion loss increases. This is because, the detector sensitivity typically increases with the increase in BR [19], and from Fig. 3.3, BR increases as the insertion loss (IL^{dB}) decreases for *PROTEUS* (circular points). Despite this increase in BR with the decrease in insertion loss for *PROTEUS*, the utilization of P_{Laser} for PP^{dB} for *PROTEUS* remains at the minimum possible value for all insertion loss values. This contrasts with what happens for OPA (Fig. 3.3). Such minimization of PP^{dB} for all insertion loss values allows for larger BR values at smaller insertion loss values for *PROTEUS*, yielding greater aggregated datarate (green columns in Fig. 3.3) for *PROTEUS* for smaller insertion loss values. Thus, dynamic adaptation of BR and Q with changing insertion loss values for each packet transfer allows *PROTEUS* to opportunistically improve the frame delay and optical power efficiency for different packet transfers.

3.5 Implementation Of *PROTEUS* Framework

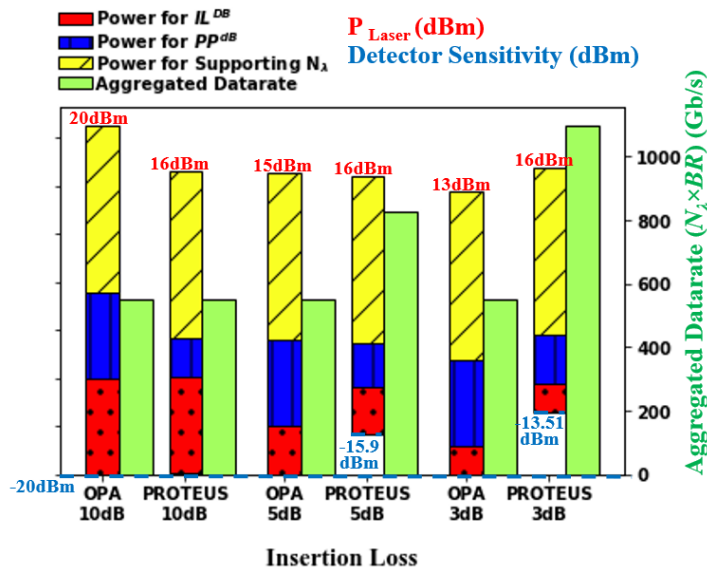
Our proposed *PROTEUS* framework uses the offline search heuristic analysis described in Section 3.4.3 to find the optimal combination of BR and Q for different IL^{dB} values. Using this offline information, *PROTEUS* dynamically adapts Q and BR to the optimum values to co-optimize optical power efficiency and frame latency, for each photonic packet transfer. *PROTEUS* incorporates a lookup table at each GI, which stores the rules in terms of the required control parameter values for adapting Q and BR . We propose to adapt Q by incorporating an MR modulator/filter design from [146], and adapt BR by implementing a light-weight reconfiguration of the serialization and deserialization modules at each GI. We discuss the operation of these adaptive designs and their incurred overheads in the next subsections. We also discuss how we derive the rules required to enable the lookup table based adaptation.

3.5.1 Dynamic Adaptation of BR

To enable dynamic reconfiguration of the BR , we propose to use reconfigurable serializer and deserializer modules at each GI, as illustrated in Fig. 3.4. In the design shown in Fig. 3.4, at each GI, the clock distribution H-tree (implementation of which is explained in Section 3.5.3) supplies a discrete set of upscaled clocks rate (10GHz, 15GHz, 20GHz and 25GHz). Each of these upscaled clock rates corresponds to the specific BR , e.g., the clock rate of 10GHz corresponds to BR of 10Gb/s. In Fig. 3.4, each GI has multiple copies of both serializer and deserializer units, with each copy enabling the clock rate scaling between the baseline rate of 5GHz (not shown in Fig. 3.4) and a specific upscaled rate. For example, the ‘5GHz to 10GHz’ (‘10GHz to 5GHz’) serializer (deserializer) unit enables clock-rate scaling from (to) the baseline value of 5GHz to (from) the upscaled value of 10GHz. The selection of the serializer and deserializer units to be used for transmission of a photonic packet is controlled by the switches S_1 , S_2 , S_3 and S_4 . The switches S_1 to S_4 are also used to gate the up-



(a)



(b)

Figure 3.3: (a) Frame Delay and Optical Power Efficiency for different insertion loss values (indicated by different colors) for OPA and *PROTEUS* (indicated with different shapes); (b) Utilization of P_{Laser} and Aggregated Datarate for OPA and *PROTEUS* across different insertion loss values.

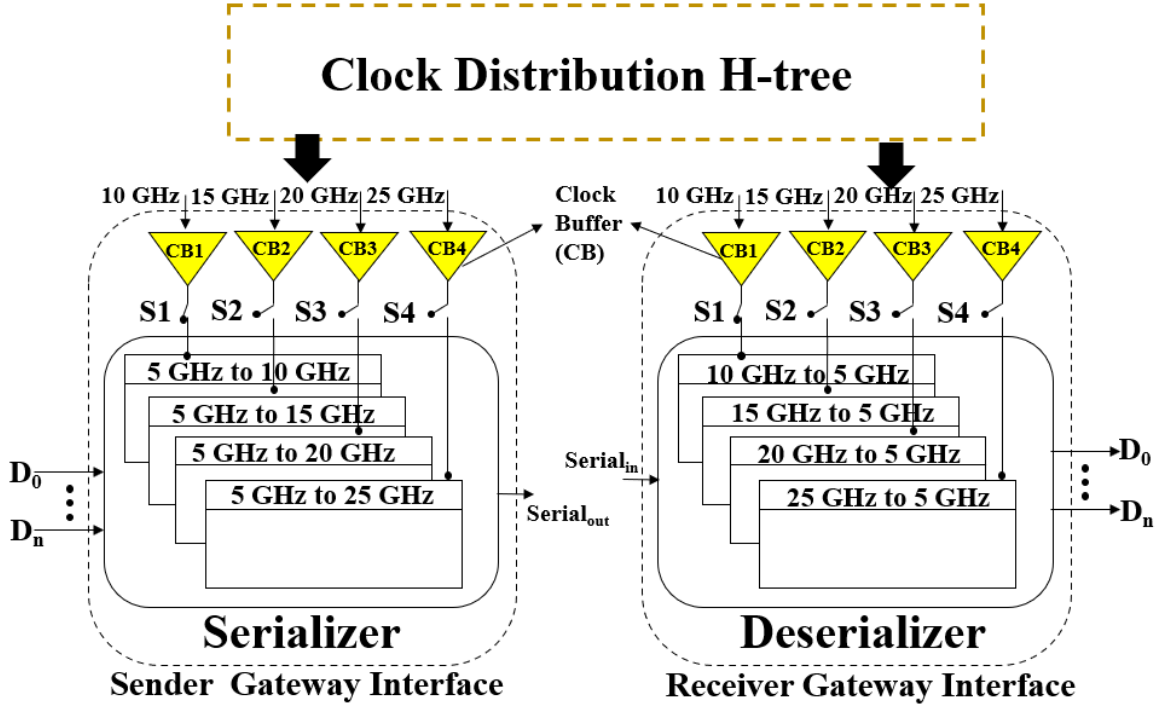


Figure 3.4: The schematic of the reconfigurable serializer and deserializer units at the sender and receiver gateway interfaces. These units along with the upscaled clock rates provided from the clock distribution H-tree and switches S_1 , S_2 , S_3 and S_4 enable dynamic adaptation BR .

scaled clock signals, so that the idle serializer and deserializer units can be turned off. For instance, in Fig. 3.4, the sender GI can serialize the input data bits (D_0 to D_n) of a packet with $BR=10\text{Gb/s}$ by configuring the switches S_1 , S_2 , S_3 , and S_4 to ‘1’, ‘0’, ‘0’, and ‘0’ states respectively, which means that switch S_1 is closed and switches S_2 to S_4 are open. These states of the switches can be collectively represented with the switch-state vector $S_1S_2S_3S_4 = '1000'$. This ‘1000’ switch-state vector basically selects the ‘5GHz to 10GHz’ serializer unit at the sender GI and the ‘10GHz to 5GHz’ deserializer unit at the receiver GI. It also gates the remaining three serializer and deserializer modules at the sender-receiver GIs to the power down mode. Thus, *PROTEUS* can use this switch-state vector $S_1S_2S_3S_4$ as the control parameter before each packet transfer at the sender and receiver GIs involved with the packet transfer, to select the appropriate serializer-deserializer pair and to consequently control the BR for the packet transfer. The overheads of this BR control mechanism are discussed next.

1. Area and Power Overhead Analysis:

The dynamic adaptation of BR incurs overhead for the generation and distribution of various upscaled clock signals. In addition, the dynamic power

overhead of the serializer and deserializer modules change with the selection of the upscaled BR . We consider the power values for the serializer and deserializer units from [154] for the 45nm CMOS SOI platform, and scale them for different upscaled BR values. Accordingly, the serializer modules corresponding to the upscaled BR values of 10Gb/s, 15Gb/s, 20Gb/s, and 25Gb/s, respectively, consume 1.4mW, 2.4mW, 3.3mW, and 4.2mW power. From [55], the deserializer units also have approximately the same power values as the serializer units. Moreover, we consider the power and area consumption of the clock generator per upscaled clock rate to be 0.5mW and $180 \mu m^2$. Further, the clock distribution H-tree also incurs similar area (for the required clock buffers across the H-tree network) and power overheads of 0.504mW and $320 \mu m^2$ per upscaled clock rate [48]. In addition, we assume that the serializer and deserializer units can be woken-up from the power-down mode in ~ 200 ps, which we think is the reasonable value as the critical path for these units can be reasonably short [115]. We include these power overhead values in our system-level simulations in Section 3.4.

3.5.2 Dynamic Tuning of Q

To enable dynamic tuning of Q , we extend the two-point coupled MZI-based MR modulator design from [146] for a faster response. Fig 3.5(a) and 3.5(b), respectively, show our utilized MR modulator and MR filter designs. In these designs, the regular coupling waveguide, which generally supports the input and through ports of the MR device, is extended to have a long coupler arm that couples with the MR at two points C1 and C2. In the original design from [146], this coupler arm is integrated with a microheater that can thermo-optically change the coupler optical path-length l_1 with respect to the MR optical path length l_2 to modulate the coefficients of light coupling at points C1 and C2, which in turn results in the modulation of quality factor (Q) for the MR. Using this method, a wide range of Q tuning has been demonstrated in [146]. However, the use of microheater results in a very slow response time for tuning Q (in the order of milliseconds). Therefore, to improve the response time, we embed a reverse-biased PN-junction based phase-shifter in the coupler arm, instead of the heater based approach in [146]. From Fig. 3.5(a), by changing the reverse bias voltage V_R across the PN-junction, the depletion layer width can be changed in the PN-junction to change the effective index of the coupler arm, which in turn can change the optical path length l_1 of the coupler arm, resulting in the change in the coupling coefficients and Q of the MR. Thus, *PROTEUS* can use the applied reverse-bias voltage across the coupler arm as the control parameter to tune the Q values for individual MR modulators and filters in the PNoC.

1. Power Overhead and Response Time Analysis:

We model our designed MRs, along with the PN-junction based phase-shifter in the coupler arms of the MRs, using the phase-shifter model given as part of the open-source modeling framework [73]. In our model, we use the nominal carrier concentration values for the P+, N+, P++, and N++ doping regions

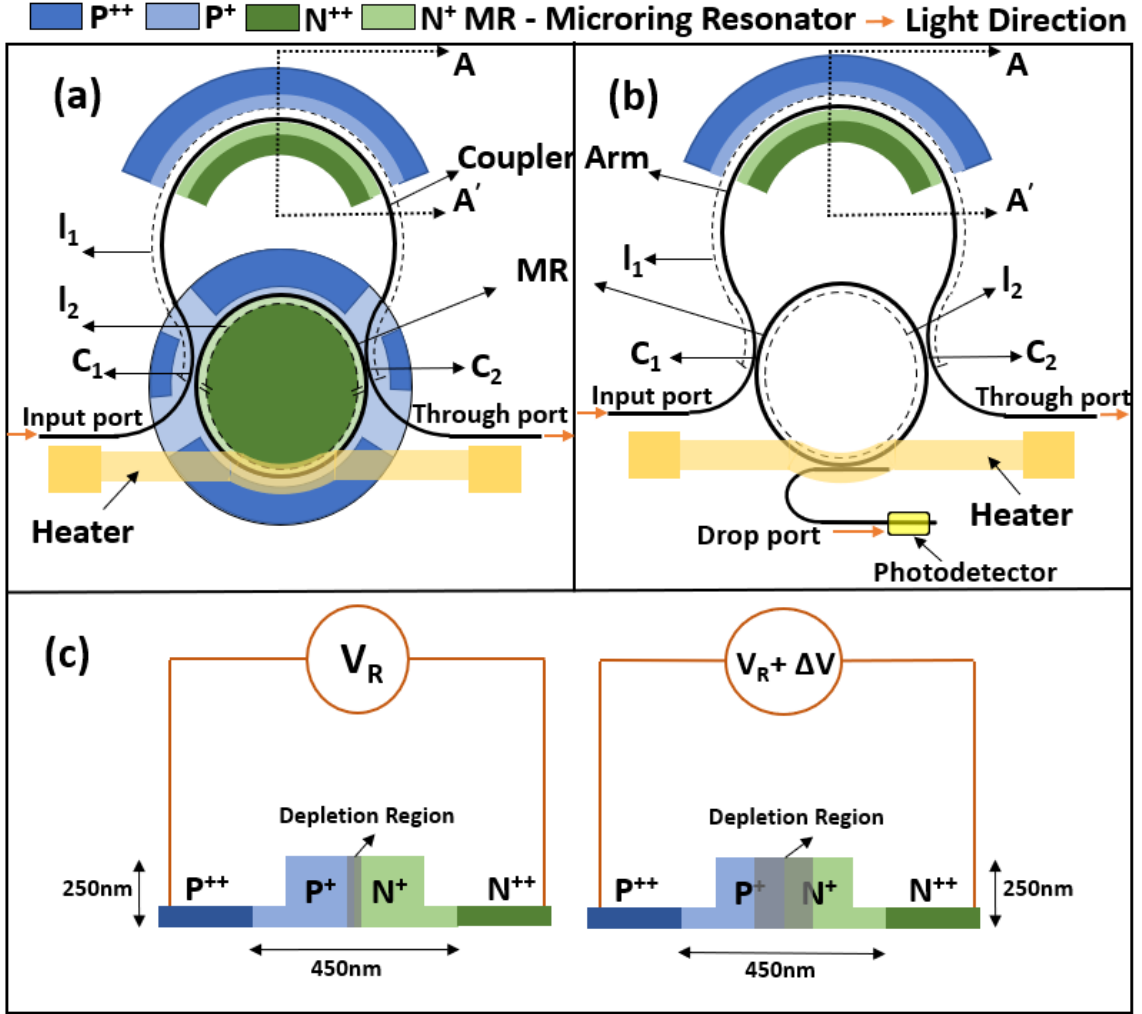


Figure 3.5: (a) Two-point coupler arm based MR modulator; (b) Two-point coupler arm based MR filter; (c) Cross-sectional view along AA' of the PN-junction embedded in the coupler arms of the MRs.

and PN-junction dimensions from [146]. From our modeling, we find that the Q of our designed MRs can be adapted with the response time to be in the range of 20-30ps.

In addition, the adaptation of Q incurs Q -tuning power overhead. Fig .3.6 gives the variation of Q with respect to the Q -tuning power, which is associated with V_R across the PN-junction. From Fig .3.6, tuning of Q values over a wide range can be achieved. From the figure, the highest Q -tuning power value is $6.1\mu W$ per MR. This value translates into total 0.03W power overhead, if the Q values for all 6457 MRs in our considered enhanced Flexishare PNoC architecture [124] are tuned.

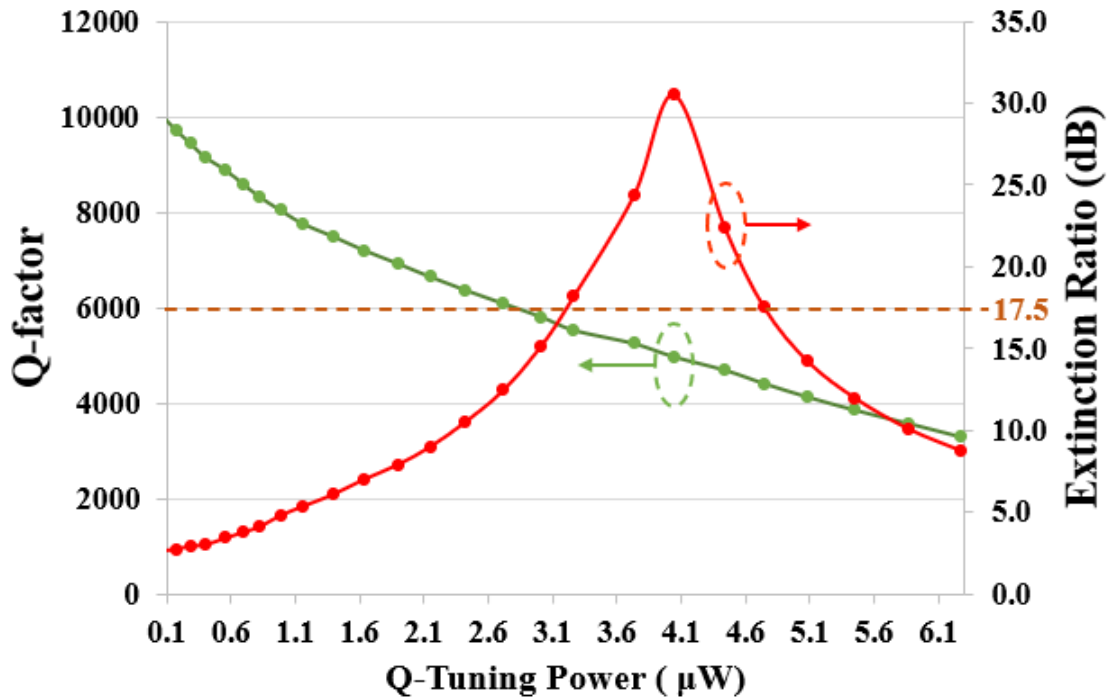


Figure 3.6: Variation of Q (Q -factor) and extinction ratio in our considered MR designs from Fig. 3.5 with applied Q -Tuning Power.

Further, Fig .3.6 also captures the dependency of the change in extinction ratio with the change in Q . This dependency results in the power penalty in MR modulators due to the limited extinction ratio of modulated signals. This power penalty can be modeled using the first term in Eq. 3.2. For that, we evaluate r from the extinction ratio value obtained from Fig .3.6. For example, the horizontal brown line shown in Fig .3.6 corresponds to $Q=6000$ and extinction ratio = 17.5dB, which in turn corresponds to $r = 10^{(17.5/10)} \approx 56.2$. Using this r value in the first term of Eq. (3.2) yields the power penalty of 0.154dB. We evaluate this power penalty as part of our offline search heuristic described in Section 3.4.3. Therefore, our selected Q and BR values for different IL^{dB} values (3.4.3) already reflect this power penalty overhead.

3.5.3 Putting All Together with Rule-Based Lookup

Fig .3.7 shows the schematic implementation of our *PROTEUS* framework. From the figure, the upscaled clock signals required for BR adaptation (not shown in Fig .3.7) are generated in the centralized clock generator, and then these clock signals are delivered to the individual GIs in the PNoC through the clock distribution H-tree. In addition, each GI in the PNoC uses an SRAM-based lookup table to stores the control parameters (i.e., the switch-state vectors $S_1S_2S_3S_4$ for BR and V_R values for Q) that enable the adaptation of BR and Q for every packet transfer. Every entry

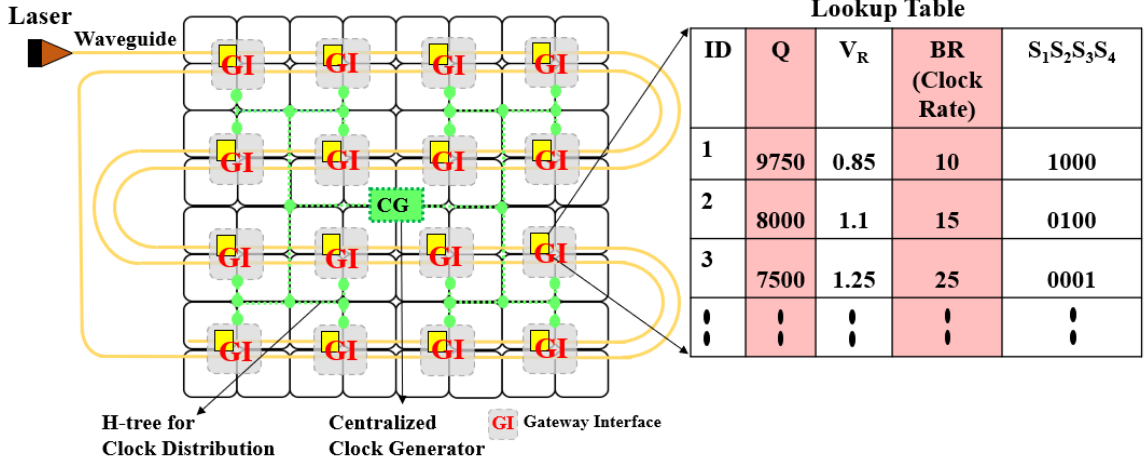


Figure 3.7: Schematic implementation of our *PROTEUS* framework on the enhanced Flexishare PNoC architecture from [35].

in the lookup table is indexed using an ID that identifies the sender-receiver GI pair to be involved for the packet transfer associated with the entry. Before each packet transfer, the associated sender and receiver GIs access the control parameters from the lookup table and adapt the BR and Q accordingly, all during the arbitration and receiver selection phases [35] that are required for successful transfers of data packets over the crossbar based PNoCs (e.g., [124], [35]).

1. Overheads of Rule-Based Lookup

The access latency of lookup table indexing is evaluated to be 40ps, using CACTI 7.0 [21] based modeling and analysis. This latency, when added to the Q -adaptation response time of 20-30ps and the wake-up time for the serializer-deserializer units of 200ps, gives the total latency for Q and BR adaptation to be 270ps, which is about half the typical processing core operating clock period of 500ps (i.e., 2GHz clock rate). Moreover, each lookup table has 64 entries (corresponding to 64 sender-receiver GI pairs) of 24-bits each, to support the storing of sender-receiver IDs, V_R values and $S_1S_2S_3S_4$ vectors. The total area overhead of all lookup tables in the PNoC is $0.09mm^2$.

3.6 Evaluation

3.6.1 Evaluation Setup

For evaluating our *PROTEUS* framework, we simulate a 256-core system with a PNoC that has 32 GIs and 32 clusters, with each cluster having 8 cores. We targeted a 22nm process node and 5 GHz clock frequency for the 256-core system. We consider the recently proposed variant [35] of the well-known Flexishare PNoC architecture [124], which employs the overlapped concurrent token stream arbitration method. Fig. 3.1 shows the physical-layer schematic of the scaled down version (i.e., 64 cores,

Table 3.1: Various Loss And Power Parameters

Parameter	Value
Laser wall-plug efficiency	10% [19]
Sensitivity at 10Gb/s	-20dBm [19]
Waveguide Insertion Loss	0.54dB/cm [31]
Waveguide Bending Loss	0.005 dB/900 [66]
Splitter Loss	0.5dB [66]
Coupler Loss	2dB [66]
Free Spectral Range (FSR)	20 nm [141]
Thermal tuning power	800 μ W/nm [155]

16 clusters, 16 GIs, 4 cores per cluster) of our considered PNoC. Our considered PNoC architecture implements intra-cluster communication in the electronic domain and inter-cluster communication in the photonic domain, as done in the PNoC from [26]. Our considered PNoC uses 32 multiple-writer-multiple-reader (MWMR) type of crossbar waveguides, with each wave-guide employing total 55 DWDM photonic signals (i.e., $N_\lambda = 55$). We consider a packet size (PS) of 512 bits, therefore, the frame delay for our PNoC becomes $|(PS/N_\lambda)| \times (1/BR) = 10/BR$. We modeled and simulated the architectures at cycle-accurate granularity with a SystemC-based in-house NOC simulator. We used real world traffic from applications in the PARSEC benchmark suite [154]. The traces of PARSEC benchmark applications were generated from gem5 full-system simulations, and then were fed into our NoC simulator. We adequately warmed up our Gem5 simulations to consequently extract the traces from the regions-of-interest (ROIs) [25] of the applications.

To compute laser power, we considered the values listed in Table 3.1 for calculating the total optical power coupled to the PNoC chip. Then, we considered the wall-plug efficiency of 10% to evaluate the electrical input power in the off-chip laser source (referred to as electrical laser power). In addition to the electrical laser power, we also evaluated the average packet latency and aggregated energy-per-bit (EPB) values. We evaluate aggregate EPB as the sum of electrical laser EPB, thermal tuning EPB, and overhead EPB. To evaluate EPB, we divide average power value with the average throughput of the PNoC, e.g., to evaluate electrical laser EPB, we divide electrical laser power with the average throughput, and vice versa. We take our overhead power values from Section 3.5, and thermal tuning power from Table 3.1.

We compared *PROTEUS* with two dynamic laser power (LP) management techniques from prior work: Adaptive Bandwidth Management technique (ABM) from [33], and On-chip Semiconductor Amplifier (OPA) based technique from [172]. ABM performs a weighted time-division multiplexing of the photonic network bandwidth and leverages the temporal fluctuations in network bandwidth to opportunistically save LP. ABM is designed to perform LP management in MWMR waveguides [33]. On the other hand, OPA uses on-chip semiconductor amplifiers to achieve traffic-independent and loss-aware savings in LP. We consider Flexishare with ABM as our base case for comparison. For comparison with ABM, it is necessary to en-

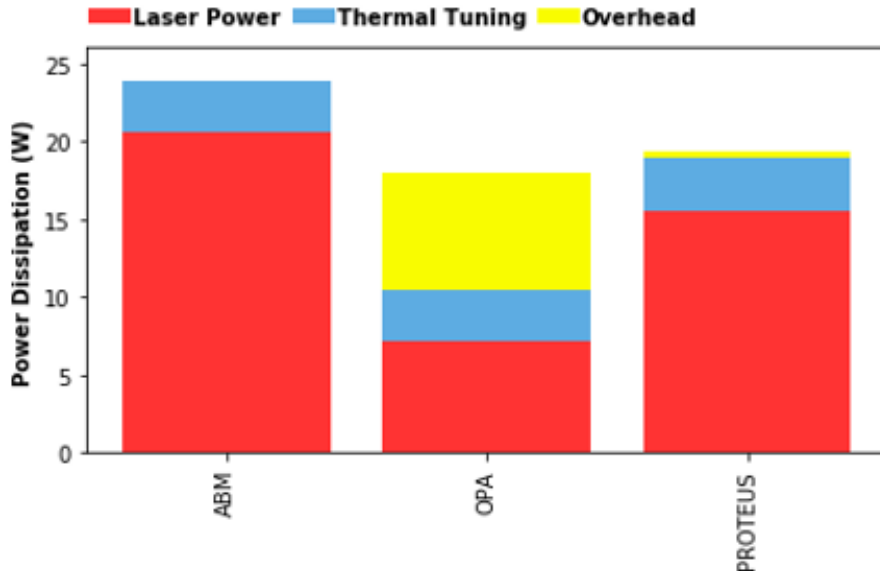


Figure 3.8: Total power (electrical laser, thermal tuning, and overhead power) dissipation results for the ABM, OPA, and *PROTEUS* enabled variants of our considered enhanced Flexishare PNoC architecture. Overhead is for adapting Q and BR .

able weighted time division multiplexing of the network bandwidth in the Flexishare PNoC. Therefore, we enhanced the Flexishare PNoC [124] with the overlapped concurrent token stream arbitration method from [35], to enable weighted time-division multiplexing of the network bandwidth. We analyzed the power dissipation, average packet latency and aggregate EPB for OPA, ABM and *PROTEUS*, when these frameworks were integrated with our considered enhanced Flexishare PNoC architecture.

3.6.2 Comparative Analysis Results

Fig .3.8 presents total power dissipation (sum of electrical laser power, thermal tuning power, and overhead power) results for ABM, OPA and *PROTEUS*. ABM does not have any power overheads involved [33], whereas OPA has the power overhead of tuning OPAs [172] and *PROTEUS* has the overhead of adapting Q and BR . Despite of this fact, the total power consumption for *PROTEUS* is less than ABM by 17.89%. This is because the static reduction in optical P_{Laser} to 16dBm for *PROTEUS* turns out to be significant than the dynamic and traffic-dependent reduction in optical P_{Laser} for ABM, which in turn reduces the electrical laser power for *PROTEUS* by 24.5%, contributing to the reduction in total power consumption. In contrast, *PROTEUS* consumes 5.13% more total power than OPA, despite OPA consuming significantly more overhead power than *PROTEUS*. This is because the optical P_{Laser} is modulated to its minimum required value for every packet transfer in OPA, which proves to be better than the static reduction in P_{Laser} achieved by *PROTEUS*, resulting in less total power consumption for OPA than *PROTEUS*. Nevertheless, *PROTEUS* achieves better average latency and EPB results, as discussed next. Fig .3.9

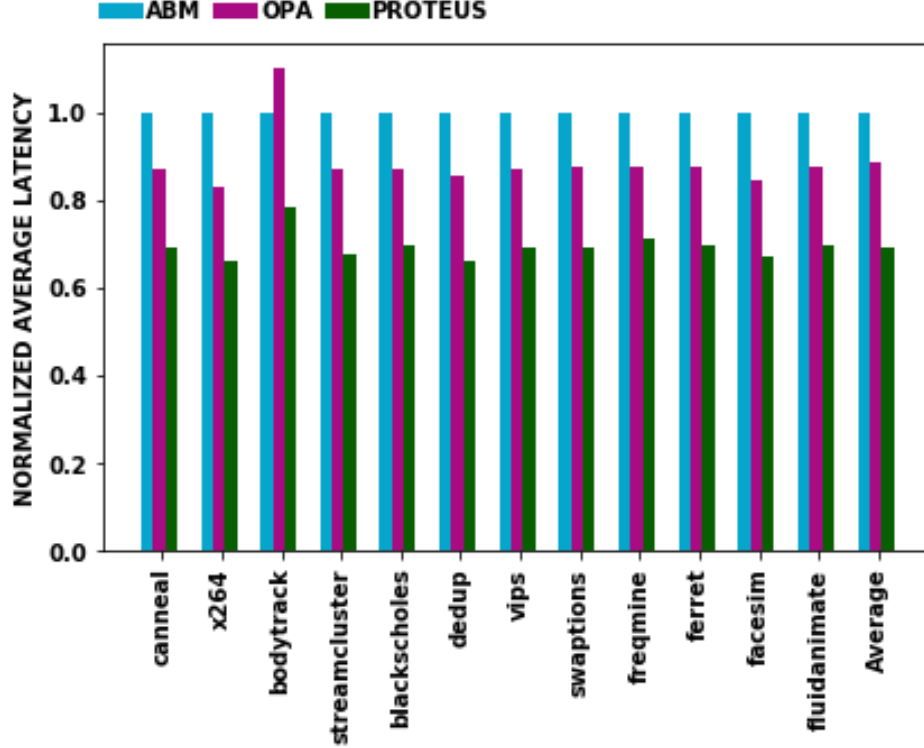


Figure 3.9: Normalized average latency for the ABM, OPA, and *PROTEUS* enabled variants of our considered enhanced Flexishare PNoC architecture. Results are normalized with respect to the ABM technique.

shows the average packet latency results, with all values normalized with respect to the ABM technique. As evident from Fig .3.8, it can be observed that on average, *PROTEUS* achieves 31% and 21.5% lower latency than ABM and OPA, respectively. The lower latency for *PROTEUS* is due to the dynamic adaptation of Q and BR , which decreases the PP^{dB} to increase the aggregated data-rate and reduce the frame delay, resulting in reduced latency. In contrast, ABM and OPA techniques do not aim to reduce average packet latency at all. Furthermore, ABM experiences higher latency compared to OPA, as ABM incurs additional latency for switching ON and OFF the off-chip laser sources [33].

Fig. 3.10 gives aggregate EPB (sum of electrical laser EPB, thermal tuning EPB, and overhead EPB) results for ABM, OPA, and *PROTEUS*. *PROTEUS* consumes 20% and 13.6% less aggregate EPB than ABM and OPA, respectively. As seen earlier, *PROTEUS* has the least average latency, which yields the highest throughput, resulting in the least EPB, compared to ABM and OPA. Moreover, as the average latency and total power for ABM are higher than OPA, ABM has greater aggregate EPB than OPA. Thus, our proposed *PROTEUS* framework is able to strike a balance between the total power consumption and performance (in terms of average packet latency) of the PNoC, and therefore, it can achieve more energy-efficiency in terms of energy-per-bit.

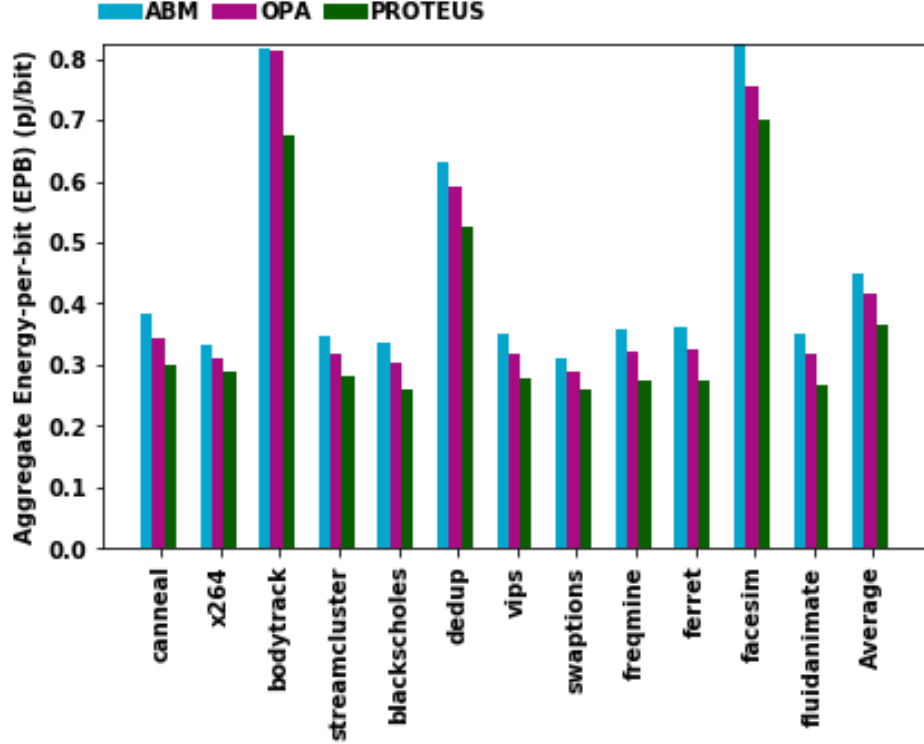


Figure 3.10: Aggregate energy-per-bit (EPB) results for the ABM, OPA, and *PROTEUS* enabled variants of our considered enhanced Flexishare PNoC architecture.

3.7 Summary

This Chapter presented an insertion loss aware framework *PROTEUS* that enables rule-based dynamic adaptation of the key photonic link configuration parameters, such as Q -factor of microrings and bitrate of photonic data signals, to statically reduce the laser power consumption and opportunistically improve the packet transfer latency in PNoCs. *PROTEUS* exploits the dependence of BER power penalty in PNoCs on Q -factor and bitrate to balance the reduction in laser power consumption in PNoCs with the achieved aggregated datarate and packet latency. Evaluation with PARSEC benchmarks shows that the *PROTEUS* framework can achieve up to 24.5% less laser power consumption, up to 31% less average packet latency, and up to 20% less energy-per-bit, compared to two other laser power management techniques from prior work. Thus, *PROTEUS* represents an attractive solution for co-optimizing the laser power consumption and performance of emerging PNoCs.

Chapter 4 Photonic Reconfigurable Accelerators for Efficient Inference of CNNs with Mixed-Sized Tensors

4.1 Introduction

Convolutional Neural Networks (CNNs) have shown record-breaking performance for implementing various real-world artificial intelligence tasks such as image recognition, language translation, and autonomous driving [202, 128, 118, 149]. The ever-increasing complexity of CNNs has pushed for highly customized CNN hardware accelerators [20]. Among them, silicon-photonic CNN accelerators have shown great promise to provide unparalleled throughput, ultra-low latency, and high energy efficiency [129, 164, 167, 109, 201, 157]. Typically, a silicon-photonic CNN accelerator consists of multiple Tensor Product Cores (TPCs) that perform multiple tensor products in parallel. Several TPC-based photonic CNN accelerators have been proposed in prior works based on various silicon-photonic devices, such as Mach Zehnder Interferometer (MZI) (e.g., [144],[131],[16]) and Microring Resonator (MRR) (e.g., [201, 157, 22],[111]).

Among these TPC-based photonic CNN accelerators from prior work, the MRR-enabled TPC-based accelerators (e.g., [166, 111, 201, 157, 22]) have shown disruptive performance and energy efficiencies for processing CNN tensor products, due to the MRRs' compact footprint, low dynamic power consumption, and compatibility with cascaded Dense-Wavelength-Division-Multiplexing (DWDM). The MRR-enabled TPCs of these accelerators transform CNN tensor products into Vector Dot Products (VDPs) by decomposing the input tensors into vectors (1D tensors). The VDP operations are performed on the individual VDP Elements (VDPEs), which are the main MRR-enabled hardware components in a TPC. Multiple VDPEs in a TPC can perform multiple VDP operations in parallel. The results of these VDP operations can be summed together (when needed) using a partial-sum (*psum*) reduction network, which can be employed outside of the TPCs as part of the post-processing components of the CNN accelerator. The functioning of the TPCs and their constituent VDPEs in the ultra-high-speed photonic domain results in disruptive throughput for processing tensors.

However, the existing MRR-enabled TPC-based accelerators are not efficient in processing modern CNNs with mixed-sized tensors, such as Xception [36] and MobileNetV1 [69]. This is because these modern CNNs utilize depthwise separable convolutions in addition to the standard convolutions. Depthwise separable convolutions in a CNN employ reduced sized tensors compared to the standard convolutions, to reduce the overall computational load of processing the CNN. For instance, MobileNetV1 shows 8-9 \times reduction in its computational load with only 1% accuracy drop [69]. But the existing MRR-enabled TPCs and their constituent VDPEs have fixed sizes. Therefore, mapping the processing of the modern CNNs with mixed-sized tensors [42] on such fixed-sized TPCs often leads to a low hardware utilization in the TPCs. This in turn diminishes the achievable performance and energy efficiency

from such fixed-sized TPCs-based accelerators. This is because the low hardware utilization incurs non-amortizable area and static power overheads while also idling away the opportunity for increasing the processing throughput.

To address this shortcoming, in this Chapter, we present a novel way of introducing reconfigurability in the MRR-enabled TPCs-based CNN accelerators, to enable efficient support for both depthwise separable convolutions and standard convolutions. To enable this reconfigurability, we invent reconfigurable VDPEs that employ MRR-based comb switches to allow re-aggregation of the CNN vectors (decomposed 1D tensors) to consequently enable dynamic resizing of the produced VDP results. Our evaluations show that our invented reconfigurable VDPEs can (1) substantially improve the hardware utilization, (2) enhance the flexibility of processing CNN tensors of various sizes, (3) improve the opportunities for parallel tensor processing, and (4) significantly enhance the energy efficiency, for CNN inference acceleration.

Our key contributions in this Chapter are summarized below:

- We review several state-of-the-art MRR-enabled TPC-based CNN accelerators from prior work and then classify the circuit-level TPC organizations used in these accelerators into two categories, namely AMM and MAM (Section 4.3.1);
- We perform the scalability analysis of TPCs of AMM and MAM categories to capture the inter-dependence of the maximum achievable TPC size, bit precision, and operating data rate (Section III-B);
- We map the processing of four state-of-the-art CNNs with mixed-sized tensors on the TPCs of AMM and MAM categories to evaluate the hardware utilization of the TPCs, to consequently establish the need for reconfigurability in the AMM and MAM-styled TPCs (Section 4.4);
- We invent a novel reconfigurable structure for VDPEs, and utilize these VDPEs to modify the AMM and MAM TPCs, to render these TPCs with the capabilities of dynamic re-aggregation of vectors for adaptive resizing of the processed VDPs (Section 4.5);
- We evaluate the performance of our designed reconfigurable MAM and AMM TPC organizations for the inference of four modern CNNs in terms of Frames-Per-Second (FPS), FPS/W, and compare it with three different AMM- and MAM-styled CNN accelerators from prior work, with area proportionate outlook for which we set the area of all our evaluated accelerators to be equal (Section 4.6).

4.2 Preliminaries

4.2.1 CNNs with Mixed-Sized Tensors

It has been established that the efficiency of executing CNNs can be improved by drastically reducing the computation, communication, and memory requirements of

the CNNs. To achieve this, there has been a growing trend of employing compressed, mixed-sized tensors in CNNs. For example, to compress the size of the utilized tensors, the Xception CNN model [36] introduced depthwise separable convolutions. Typically, a Depthwise Separable Convolution (DSC) breaks up a Standard Convolution (SC) into a Depthwise Convolution (DC) and a subsequent Pointwise Convolution (PC). Fig. 4.1 demonstrates how a Standard Convolution (SC), Depthwise Convolution (DC), and Pointwise Convolution (PC) work. An SC performs the channel-wise and spatial-wise tensor product computation in a single step by applying a single 3D kernel (convolutional filter) tensor to all the channels of the input tensor. In contrast, a DSC splits the tensor product computation into two steps. In the first step, the constituent DC applies a dedicated 2D kernel tensor per channel of the input tensor, and the channel-wise outputs are stacked to produce a single intermediate tensor. Then, in the second step, the subsequent PC is used to create a linear combination of all the channels of the intermediate tensor by applying a 1D kernel tensor for each spatial point of the intermediate tensor, to consequently produce the final output tensor. This can be better understood by referring to Fig. 4.1, as explained next.

Standard Convolutions (SCs)

From Fig. 4.1(a), in SC, the input tensor is $H \times W \times D$ in size, where H is the height, W is the width and D is the number of channels (i.e., the depth). The kernel tensor has dimensions $K \times K \times D$, which is convolved over the input tensor to generate a single output tensor with dimensions $H^{Out} \times W^{Out} \times 1$. If there are F such kernel tensors convolved over the input tensor, then the output tensor will have the dimensions $H^{Out} \times W^{Out} \times F$ (Fig. 4.1(a) shows $F=1$). If we define the tensor as a multidimensional array of points, then each point in the output tensor is obtained by performing a tensor product (sum of point-wise products) of the $K \times K \times D$ kernel tensor and the corresponding $K \times K \times D$ part of the input tensor (part of the input tensor highlighted in gray in Fig. 4.1(a)).

Depthwise Separable Convolutions (DSCs)

As mentioned earlier, each DSC is typically split into a DC and a subsequent PC. From Fig. 4.1(b), in DC, each channel of the input tensor has a corresponding 2D kernel tensor with dimensions $K \times K \times 1$. Since there are D channels in the input tensor, there are D such kernel tensors. Convolution of these channel-wise kernel tensors across the spatial dimensions of their respective input channels generate a total of D channels of the intermediate tensor with each channel being of dimensions $H^{Out} \times W^{Out} \times 1$. Subsequently, in PC (Fig. 4.1(c)), the intermediate tensor is convolved with a pointwise kernel tensor of dimensions $1 \times 1 \times D$ to generate one output tensor of dimensions $H^{Out} \times W^{Out} \times 1$. The use of DSCs (DCs+PCs) results in a reduction in the required number of weight points and point-wise multiplication operations [69]. Because of this advantage, several state-of-the-art CNN models, such as EfficientNet [170], ShuffleNetV2 [208] and MobileNetV2 [138], adopt DSCs.

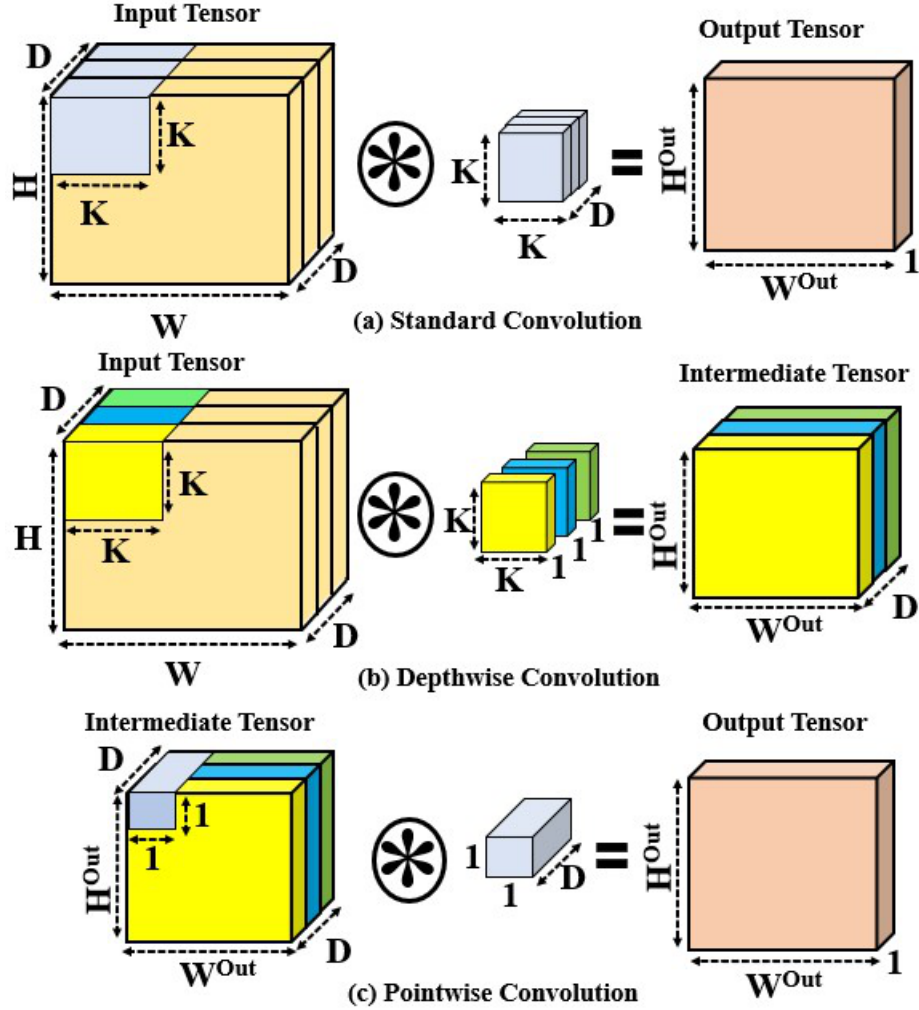


Figure 4.1: Illustration of various types of convolutions.

4.2.2 Accelerating CNN Tensor Products

Modern CNNs employ a large number of input and kernel tensors across all their constituent standard convolutional, depthwise separable convolutional and fully-connected layers. These CNN tensors are often irregular in size. For example, ResNet50[65] (EfficientNetB7[170]) CNN employs kernel tensors of at least 12 (26) different sizes (see the sizes in Table 4.3). Accelerating the inference of such CNNs having mixed-sized tensors requires efficient hardware implementations of the Products (a.k.a. General Matrix Multiplications (GEMMs)) of their input and kernel (weight) tensors. To make the hardware implementations of such CNN Tensor Products feasible, the involved input and kernel tensors are typically decomposed into vectors (1D tensors). These vectors are referred to as Decomposed Input Vectors (DIVs) and Decomposed Kernel Vectors (DKVs), henceforth. Decomposing into DIVs and DKVs in turn transforms the Tensor Products into decomposed vector dot products (VDPs), which can be efficiently performed on VDP hardware accelerators [144]. When implemented

on hardware, these decomposed VDPs produce partial results (*psums*), which can then be post-processed to achieve the final Tensor Product results using the *psum* reduction networks employed in the hardware accelerators [144].

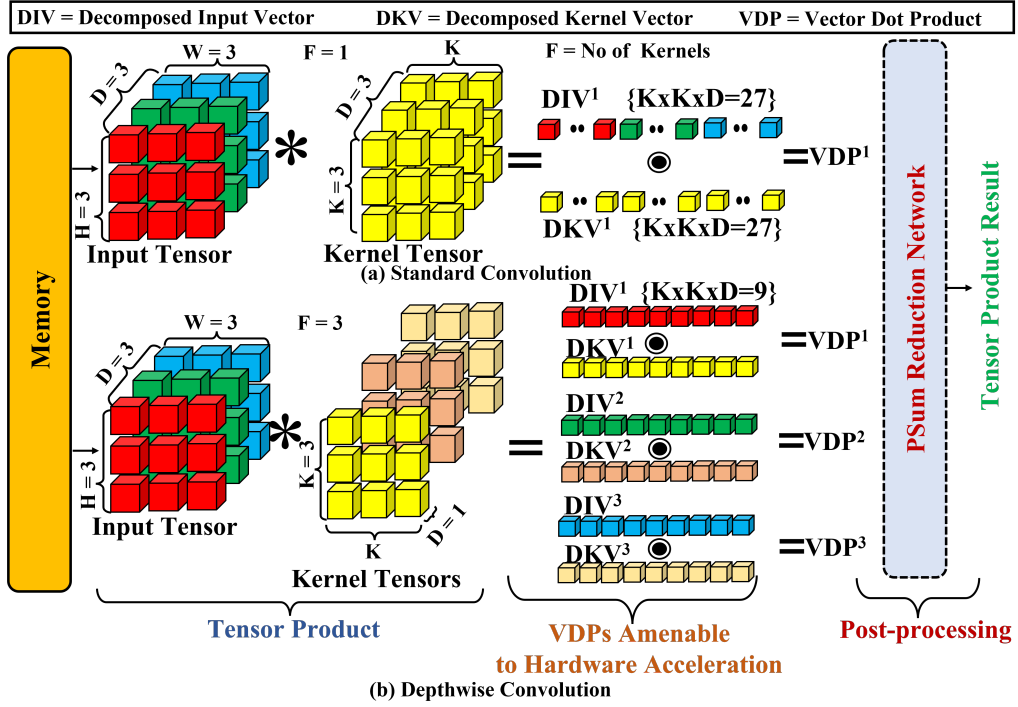


Figure 4.2: Example decomposition of CNN tensors for tensor product acceleration, for (a) standard convolution (SC), and (b) depthwise convolution (DC).

To conceive this process of Tensor Product acceleration, consider Fig. 4.2, which illustrates tensor products between the input and kernel tensors, for a SC (Fig. 4.2(a)) and a DC (Fig. 4.2(b)). In Fig 4.2(a), the tensor product between an input tensor of dimension $(H, W, D) = (3, 3, 3)$ and a single kernel tensor $(F = 1)$ of dimension $(K, K, D) = (3, 3, 3)$ is illustrated. To make this tensor product amenable to hardware acceleration, these input and kernel tensors are respectively flattened into a DIV (DIV^1) and a DKV (DKV^1) of 27 points each (corresponding to $K \times K \times D = 3 \times 3 \times 3$). Consequently, the tensor product is converted into a VDP operation between the DIV^1 and DKV^1 , which produces VDP^1 as the result. Producing this VDP^1 can be simplified into 27 point-wise multiplication operations to produce 27 point-wise products, which are then summed together using 27 accumulation operations. This makes producing VDP^1 amenable to acceleration on any computing hardware that can support multiple parallel multiplications and accumulations (e.g., [114], [79], [148], [204], [117]). If the number of supported, in-parallel multiplications and accumulations in the employed accelerator hardware is less than the size of the VDP operation (27 in our example), the result VDP^1 has to be decomposed into multiple *psum* results. These *psum* results are then summed together using the *psum* reduction network (Fig. 4.2), to produce the final tensor product result, at the cost

of extra latency. Note that for the example illustrated in Fig. 4.2, we assume that the accelerator hardware size matches with the VDP operation size.

In contrast, as discussed earlier, performing a DC operation on the same input tensor of dimension $(H, W, D)=(3, 3, 3)$ requires 3 channel-wise kernel tensors ($F=3$) of dimension $(K, K, D) = (3, 3, 1)$ each. The need to use 3 channel-wise kernel tensors necessitates that a total of 3 tensor products are obtained. For that, as shown in Fig. 4.2(b), the 3 channels of the input tensor are flattened into 3 DIVs (DIV^1, DIV^2, DIV^3), and the 3 channel-wise kernel tensors are flattened into 3 DKVs (DKV^1, DKV^2, DKV^3), of 9 points each (corresponding to $K \times K \times D = 3 \times 3 \times 1$). Performing VDP operations between respective DIVs and DKVs produces 3 VDP results (VDP^1, VDP^2, VDP^3). Thus, compared to an SC operation, a DC operation renders smaller sized DIVs and DKVs. This in turn reduces the parallelism requirement per VDP result in the employed accelerator hardware, because implementing the VDP operations between the reduced sized DIVs and DKVs requires the hardware support for a less number of in-parallel multiplications and accumulations per VDP result (only 9 in our example of DC). If the employed accelerator hardware supports more parallelism than necessary for implementing a DC operation, it can lead to lower hardware utilization efficiency. Similar to the SC and DC operations, it is also common to convert a PC operation into multiple VDP operations to make it amenable to hardware acceleration. *In summary*, to make processing of CNN tensor products amenable to hardware-based acceleration, this process of tensor flattening and decomposition remains consistent for the input and kernel tensors of arbitrary sizes (for sizes other than considered in Fig. 4.2 as well), across all types of convolutional and fully-connected CNN layers.

4.2.3 Related Work on Photonic CNN Accelerators

To accelerate machine learning tasks with low latency and low energy consumption, prior works have proposed various accelerators based on Photonic Integrated Circuits (PICs) (e.g., [20, 140, 116, 7, 144, 16, 206, 187]). Among these, the CNN accelerators employ PIC-based TPCs to perform CNN tensor products. Some accelerators implement digital TPCs (e.g., [34, 94]), whereas some others employ analog TPCs (e.g., [144, 38, 20, 166]). Different TPC implementations employ MRRs (e.g., [116, 7, 144, 111, 51]) or MZIs (e.g., [163, 16, 206]) or both (e.g., [34]). The analog TPCs can be further classified as incoherent (e.g., [144, 116, 20]) or coherent (e.g., [163, 63, 211, 199, 210, 107, 213]). To set and update the values of the individual input and kernel tensors used for tensor products, the incoherent TPCs utilize the analog optical signal power, whereas the coherent TPCs utilize the electrical field amplitude and phase. The coherent TPCs achieve low inference latency, but they suffer from control complexity, high area overhead, low scalability, low flexibility, high encoding noise, and phase error accumulation issues [164]. In contrast, the incoherent TPCs based accelerators achieve better scalability and lower footprint, because they use MRR-based compact PICs [144], unlike the coherent TPCs that use MZI-based bulky PICs.

Various state-of-the-art photonic CNN accelerators are well discussed in a survey

paper [211]. Because of the inherent advantages of MRR-enabled incoherent TPCs, there is impetus to design more energy-efficient and scalable CNN accelerators employing MRR-enabled incoherent TPCs. However, the CNN accelerators from prior works have mainly focused on designing their constituent TPCs only for the standard convolutional layers of CNNs. Prior works have paid very little attention to accelerate the processing of the depthwise separable convolutional layers of CNNs. In this Chapter, we contribute towards making the MRR-enabled incoherent TPCs more efficient by enabling them to dynamically adapt to process the standard convolutional and depthwise separable convolutional layers of CNNs.

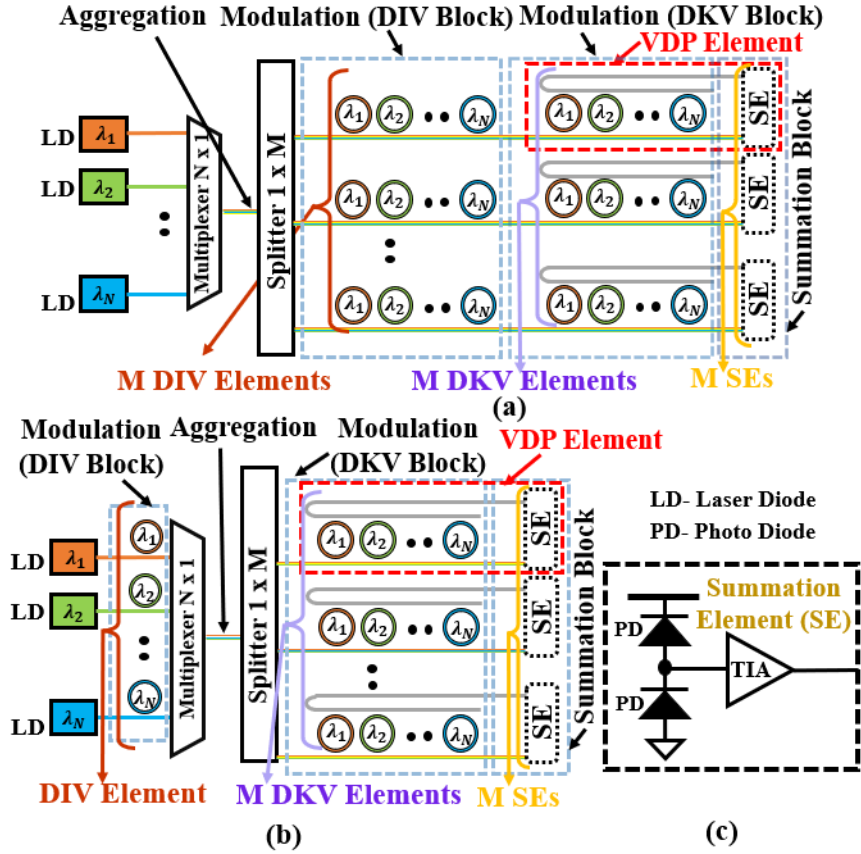


Figure 4.3: Illustration of common TPC organizations. (a) AMM organization, (b) MAM organization, and (c) Summation Element.

4.3 Classification and Scalability Analysis of TPC Organizations

4.3.1 Classification

Most of the photonic MRR-enabled analog, incoherent CNN accelerators from prior works employ multiple analog TPCs that work in parallel. Typically, an analog TPC implements the decomposed VDP operations of a tensor product (Fig. 4.2; Section 4.2.2). For that, the TPC typically employs a total of five blocks (Fig. 4.3(a)): (i)

Table 4.1: List of abbreviations and their full forms used in this Chapter. Definition and values of various parameters (obtained from [22]) used in Eq. 4.1, Eq. 4.2, and Eq. 4.3 for the scalability analysis of AMM and MAM TPCs.

Abbreviations	Full form	Parameter	Definition	Value
TPC	Tensor Processing Core	P_{Laser}	Laser Power Intensity	10 dBm
DSC	Depthwise Separable Convolution	R	PD Responsivity	1.2 A/W
SC	Standard Convolution	R_L	Load Resistance	50 Ω
PC	Pointwise Convolution	I_d	Dark Current	35 nA
DC	Depthwise Convolution	T	Absolute Temperature	300 K
DKV	Decomposed Kernel Vector	DR	Data Rate	10 GS/s
DIV	Decomposed Input Vector	RIN	Relative Intensity Noise	-140 dB/Hz
VDP	Vector Dot Product	η_{WPE}	Wall Plug Efficiency	0.1
S	Size of DKV	IL_{SMF} (dB)	Single Mode Fiber Insertion Loss	0
DKV_S	DKV of size S	IL_{EC} (dB)	Fiber to Chip Coupling Insertion Loss	1.6
F_S	Set of DKVs with size S	IL_{WG} (dB/mm)	Silicon Waveguide Insertion Loss	0.3
DR	Data rate	$EL_{Splitter}$ (dB)	Splitter Insertion Loss	0.01
VDPE	Vector Dot Product Element	IL_{MRM} (dB)	Microring Modulator (MRM) Insertion Loss	4
N	Size of VDPE	OBL_{MRM} (dB)	Out of Band Loss MRM	0.01
M	Number of VDPEs per TPC Unit	IL_{MRR} (dB)	Microring Resonator (MRR) Insertion Loss	0.01
SE	Summation Element	$IL_{penalty}$ (dB) (MAM)	Network Penalty	4.8
CS	Comb Switch	$IL_{penalty}$ (dB) (AMM)	Network Penalty	5.8
y	Number of Comb Switches	d_{MRR}	Gap between two adjacent MRRs	20 μm
x	Re-aggregating or Filtering wavelength count	$d_{element}$ (MAM)	Thermal isolation spacing between DIV and DKV elements	0 μm
L	Set of re-aggregated wavelengths	$d_{element}$ (AMM)		100 μm

a laser block that employs N Laser Diodes (LDs) to generate N optical wavelength channels; *(ii)* an aggregation block that aggregates the generated optical wavelength channels into a single photonic waveguide through DWDM (using an $N \times 1$ multiplexer) and then splits the optical power of these N wavelength channels equally into M separate waveguides (using a $1 \times M$ splitter); *(iii)* a modulation block, also referred to as DIV block, that employs M arrays of MRRs (one array per waveguide, with each array having N MRRs; each array referred to as DIV element) to imprint M DIVs of N points each onto the $N \times M$ wavelength channels by modulating the analog power amplitudes of the wavelength channels; *(iv)* another modulation block, referred to as DKV block, that employs another M arrays of MRRs (one array per waveguide, with each array having N MRRs; each array referred to as DKV element) to further modulate the $N \times M$ wavelength channels with DKVs, so that the analog power amplitudes of the individual wavelength channels then represent the point-wise products of the utilized DKVs and DIVs; and *(v)* a Summation Block (SB) that employs a total of M Summation Elements (SEs), with each SE having two balanced Photodiodes (PDs) upon which the point-wise-product-modulated N wavelength channels are incident to produce the output current that is proportional to the VDP result (i.e., the sum of the N input point-wise products). The laser block and SB are typically positioned at the two ends of the TPC, with the aggregation, modulation (DIV), and modulation (DKV) blocks placed in between them.

Based on the order in which these intermediate blocks (aggregation, modulation (DIV), modulation (DKV) blocks) are positioned between the laser block and SB, we classify the MRR-based TPC organizations from prior work as MAM (Modulation, Aggregation, Modulation) [144] or AMM (Aggregation, Modulation, Modulation) [140]. Fig. 4.3 illustrates MAM and AMM TPC organizations. From Fig. 4.3(a), the AMM TPC organization positions the aggregation block first, and then the DIV block

followed by the DKV block. In contrast, the MAM TPC in Fig. 4.3(b) positions the DIV block first, and then positions the aggregation block followed by the DKV block. Note that the MAM-styled DIV block is structurally different from the AMM-styled DIV block. The MAM-styled DIV block employs only one MRR per waveguide, and as a result, it can imprint only 1 DIV of N points onto the N wavelength channels. This 1 DIV is shared among all DKVs in the MAM TPC, whereas each DKV can have a different DIV corresponding to it in the AMM TPC.

Moreover, note that each DKV element in both MAM and AMM TPCs have two waveguides (shown but not labelled in the figures). First, the drop waveguide, coupling with the MRRs at the top. Second, the through waveguide, coupling with the MRRs at the bottom. The wavelengths that carry negatively-signed pointwise products have more guided optical power in the drop waveguide, whereas the wavelengths that carry positively-signed pointwise products have more guided optical power in the through waveguide. The drop waveguide makes its guided power incident upon the top-side PD in the corresponding SE, whereas the through waveguide makes its guided power incident upon the bottom-side PD. This enables a signed accumulation of the pointwise products carried by the guided wavelengths, because the top-side and bottom-side PDs in the SE are balanced [140]. In both the AMM and MAM TPC organizations, we refer to the combination of a DKV element and the corresponding SE as VDP element (VDPE). A VDPE size (i.e., N) should match the DKV size for efficient, low-overhead implementation of the VDP operation.

4.3.2 Scalability Analysis

Prior works [22] and [94] have shown that the scalability of photonic accelerator architectures, in terms of the achievable VDPE size N , decreases with the increase in the required bit precision. However, these prior works lack in two ways. First, they do not capture the impact of the utilized data rate on the inter-dependence between N and bit precision. Second, they do not provide the scalability analysis for AMM TPC architectures (they only analyze MAM TPCs). To address these shortcomings, we extend the methodology from [22] to perform the scalability analysis of the AMM and MAM TPCs to capture the inter-dependence of VDPE size N , number of VDPEs per TPC M , bit precision, and data rate. From [22], it is known that the bit precision of an MRR-based VDPE depends on the output photodetector sensitivity (P_{PD-opt}) and data rate (DR) [22]. In this Chapter, we evaluate the required P_{PD-opt} for various bit precision values and DRs by solving Eq. 4.1 [22]. We sweep for DRs of 1, 3, 5, and 10 GS/s (Giga-Symbols per sec), and sweep for bit precision values of 1-bit to 8-bit, and consider $M = N$ for our analysis. The value of N is obtained by solving Eq. 4.3 [22], along with P_{PD-opt} for a given bit precision and DR obtained from Eq. 4.1 and Eq. 4.2 [22].

Table 4.1 reports the definitions of the parameters and their values from Eq. 4.1, Eq. 4.2, and Eq. 4.3, as used for our analysis. In Table 4.1, $P_{penalty}$ represents the impairments due to extinction ratio, crosstalk, inter-symbol interference (ISI), and laser relative intensity noise (RIN). We perform this analysis for both AMM and MAM TPC architectures. As discussed in section 6.3, AMM and MAM TPCs

differ in the placement of the DKV and DIV elements. In MAM TPCs, all DKV elements share a single DIV element, whereas in AMM TPCs all DKV elements have their individual DIV elements. Therefore, to avoid thermal crosstalk in AMM TPCs, the DKV elements need to be placed sufficiently farther from their respective DIV elements [34], which in turn increases the required $d_{element}$ for AMM TPCs (Table 4.1), thereby increasing the optical lengths of the waveguides in AMM TPCs. Due to the longer waveguides, the $IL_{penalty}$ increases for AMM TPCs compared to MAM TPCs (Table 4.1). This in turn affects the achievable VDPE size N , DR, and bit precision for AMM TPCs, as confirmed by our below-presented analysis results.

$$n_{i/p} = \frac{1}{6.02} \left[20 \log_{10} \left(\frac{R \times P_{PD-opt}}{\beta \sqrt{\frac{DR}{\sqrt{2}}}} - 1.76 \right) \right] \quad (4.1)$$

$$\beta = \sqrt{2q(RP_{PD-opt} + I_d) + \frac{4kT}{R_L} + R^2 P_{PD-opt}^2 RIN} \quad (4.2)$$

$$P_{Laser} = \frac{10^{\frac{\eta_{WG}(dB)[N(d_{MRR})+d_{element}]}{10}} M}{\eta_{SMF} \eta_{EC} IL_{i/p-MRM}} \times \frac{P_{PD-opt}}{\eta_{WPE} IL_{MRR}} \times \frac{1}{(OBL_{MRM})^{N-1} (EL_{splitter})^{\log_2 M}} \times \frac{1}{(OBL_{MRR})^{N-1} (IL_{penalty})} \quad (4.3)$$

Fig. 4.4 and Fig. 4.5 show the inter-dependence of VDPE size N , bit precision, and DR for MAM and AMM TPCs respectively. From Fig. 4.4, for MAM TPCs, the maximum VDPE size N significantly decreases from $N = 159$ for 1-bit precision to $N = 1$ for 8-bit precision. Similarly, from Fig. 4.5, for AMM TPCs, the supported N drops from $N = 99$ for 1-bit precision to $N = 1$ for 8-bit precision. This trend of decreasing N with increasing bit precision is in line with the similar trend observed in recent work Albireo [94], although note that the accelerator architecture of Albireo is different from our considered AMM and MAM architectures. Moreover, we can also observe from Fig. 4.4 and Fig. 4.5 that the supported N also varies with DR. With the increase in DR, the supported N decreases in both MAM and AMM TPCs. For instance, MAM TPCs' supported N for 4-bit precision drops from $N = 44$ at 1 GS/s to $N = 16$ at 10 GS/s. In case of AMM TPCs, at 4-bit precision, the supported N drops from $N = 31$ at 1 GS/s to $N = 12$ at 10 GS/s. For given bit precision and DR values, AMM TPCs support lower N compared to MAM TPCs. This is because, as discussed earlier, AMM TPCs incur higher $IL_{penalty}$. From our analysis, it can be inferred that AMM and MAM TPCs cannot support any N for 8-bit precision; therefore, we advocate that AMM and MAM TPCs should be utilized to achieve 4-bit precision at the highest, to select such a power-of-two precision value below the unattainable 8-bit precision that can support a tangible N value. Our obtained values of N for different DRs are given in Table 4.2.

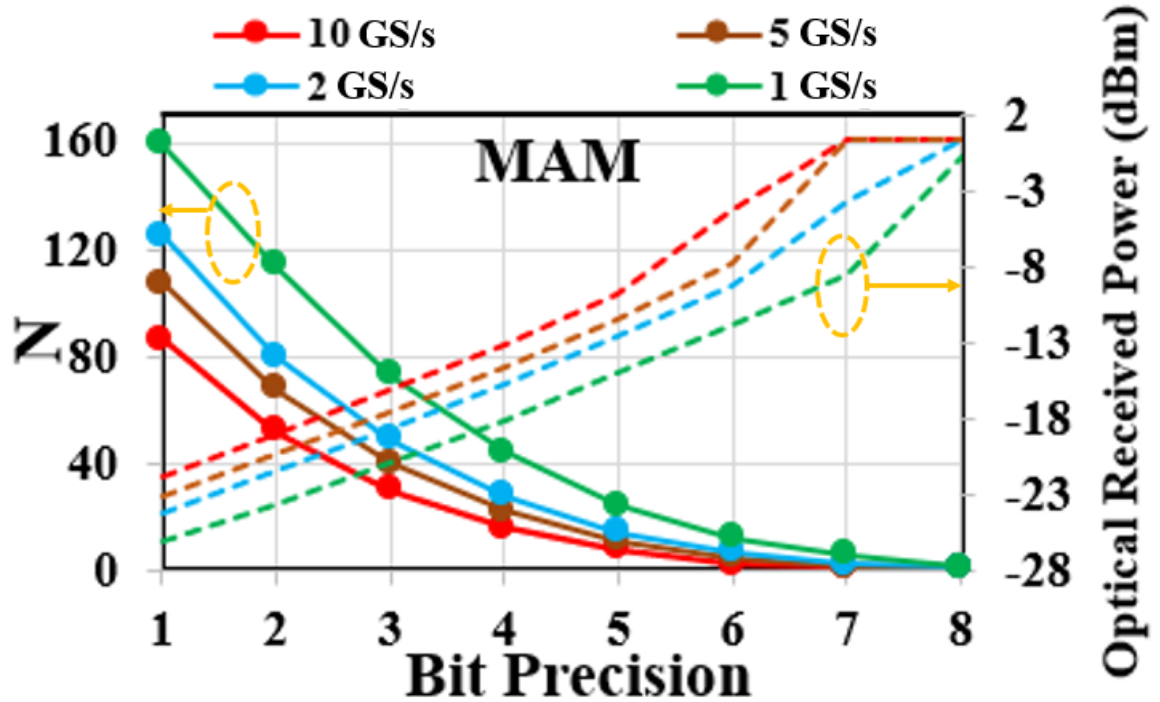


Figure 4.4: Supported VDPE size N and the optical received power (dBm) for bit precision = {1, 2, 3, 4, 5, 6, 7, 8}bits at data rates (DRs) = {1, 3, 5, 10}GS/s, for MAM TPCs.

Table 4.2: VDPE size (N) at 4-bit precision across various DRs for different TPC architectures.

TPC Architectures	DR (GS/s)			
	1	3	5	10
RMAM	43	27	22	16
RAMM	31	20	16	12
MAM (HOLYLIGHT [144])	44	28	22	16
AMM (DEAPCNN [140])	31	20	16	12

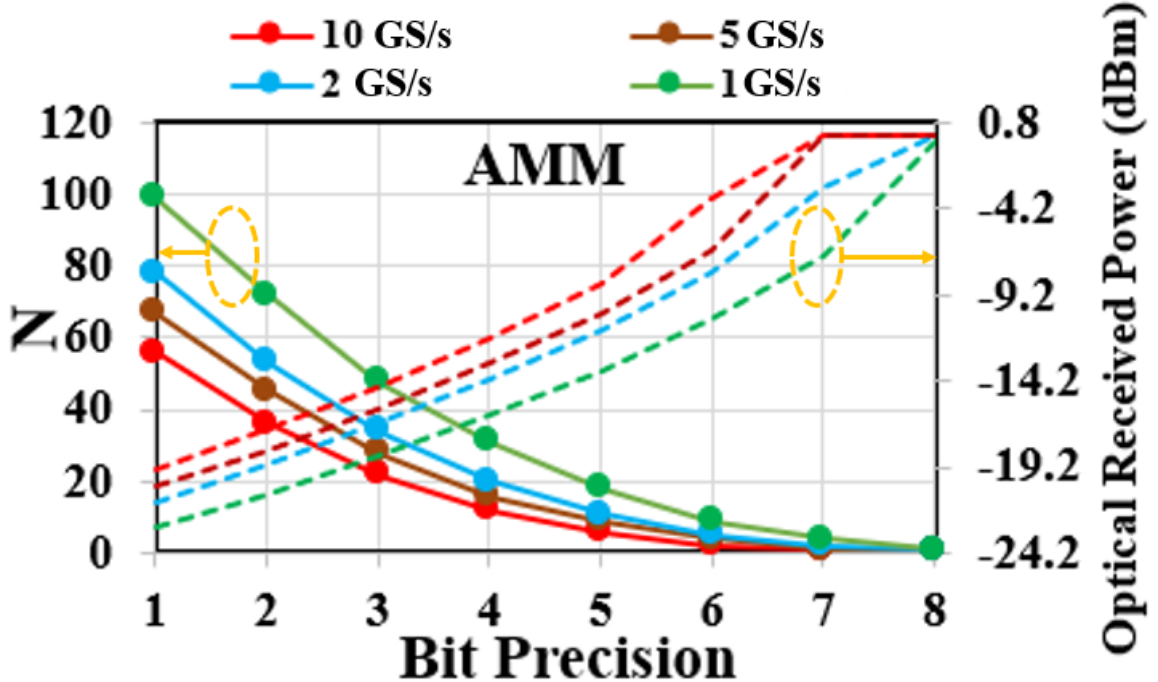


Figure 4.5: Supported VDPE size N and the optical received power (dBm) for bit precision = {1, 2, 3, 4, 5, 6, 7, 8}bits at data rates (DRs) = {1, 3, 5, 10}GS/s, for AMM TPCs.

4.4 Need for Reconfigurability in TPCs

A feasible acceleration of CNN tensor products on MRR-enabled incoherent, analog TPCs mandates that a CNN kernel tensor of shape (K, K, D) is decomposed/flattened into a 1D DKV of size $S = K \times K \times D$. From Fig. 4.2, the corresponding DIV should also be of size S . In modern CNNs, the value S corresponding to various kernel tensors vary drastically. Table 4.3 provides kernel tensor shapes and corresponding DKV sizes (S) for EfficientNetB7 [170]. We selected EfficientNetB7 as an example of CNNs with a large number of DCs, PCs, and SCs. From Table 4.3, DCs have a small set of DKV sizes ($S \in \{9, 25\}$). In contrast, in Table 4.3, PCs have a wide range of S values, from as small as 8 to as large as 3840. As discussed earlier (Section II-B), for the processing of PCs and DCs using TPCs to produce the final tensor products, their corresponding DKVs are mapped onto the constituent VDPEs of the TPCs so that the tensor products are converted into VDP operations. Therefore, depending on how the size S of a DKV compares with the size N of a VDPE, the following three scenarios arise for the mapping of the DKV onto the VDPE to produce the final VDP result:

- **Scenario 1, $S=N$:** For this case, all the DKV points have one-to-one mapping with all the MRRs of the VDPE, and there are no idle MRRs in the VDPE. The VDP result will be the final tensor product result.

- **Scenario 2, $S > N$:** In this case, a single VDPE cannot produce the final tensor product result, as it cannot process the entire DKV. Therefore, the DKV needs to be further decomposed into a total P partial DKVs requiring a total of $P = \text{Ceil}(S/N)$ VDPEs to produce the final tensor product result. All P VDPEs produce partial VDP results called *psums*, and during post-processing, a reduction network accumulates these *psums* to generate the final tensor product result. Although this accumulation of *psums* incurs additional latency, which can be efficiently hidden by employing a pipelined design of the *psum* reduction network with non-blocking bandwidth [93]. One drawback, however, is that if S is not an integer multiple of N , then this scenario leads to some unutilized MRRs in the VDPEs across P partial VDP operations.
- **Scenario 3, $S < N$:** In this case, the result of the single VDP operation provides the final tensor product result, but some MRRs in the VDPE remain unutilized. The count of unutilized MRRs depends on the size difference between the DKV and VDPE (i.e., between S and N).

For the last two scenarios (for which $S \neq N$), the unutilized MRRs cause under-utilization in the VDPEs. This hampers the performance and efficiency of processing modern CNNs with mixed-sized tensors. This is because the unutilized MRRs incur area and static power overheads while also idling away the opportunity for increasing the processing throughput. Therefore, how well N matches with S plays a crucial role in determining the performance and efficiency of a TPC. To that end, we reason that dynamic flexibility in the supported value of N in the VDPEs is required to efficiently support processing of various sizes of DCs, PCs, and SCs of modern CNNs.

4.4.1 Perils of Fixed VDPE Size (N) in TPCs from Prior Work

To validate our reasoning presented just before this subsection, we evaluated the hardware utilization values for various TPC architectures (with fixed N) from prior works. For example, MAM (HOLYLIGHT [144]), AMM (DEAPCNN [140]) have $N=43$ and $N=31$, respectively. For these TPCs, Fig. 4.6 shows hardware (MRR) utilization per-VDPE in terms of the ratio (in %) of the utilized VDPE area over the total (utilized+idle) area. The figure also shows the utilization for our proposed Reconfigurable MAM (RMAM) and Reconfigurable AMM (RAMM) TPCs. But we introduce and discuss our RAMM and RMAM TPCs in Section 4.5, so please look past their excellent utilization results for now. From Fig. 4.6, MAM (HOLYLIGHT [144]), and AMM (DEAPCNN [140]) yield significantly low per-VDPE utilization (as low as 8%). This is because of the huge mismatch between N and S (i.e., $S < N$) while processing DCs and PCs. Such low VDPE utilization can hamper performance and efficiency of the TPCs, as discussed earlier.

These results motivate the need to design a VDPE that can dynamically adapt to various DKV sizes. Therefore, we invented a novel reconfigurable VDPE design, which is described in the next section.

Table 4.3: Kernel tensor shapes (K, K, D), the total number of such kernels (F) and corresponding DKV sizes (S) for EfficientNet_B7[170], as an example CNN with a large number of DSCs. (FC=Fully Connected Layer and other abbreviations are defined in Table 4.1). The K, D, F values were extracted from Keras Applications [37].

Model	Convolution	Tensor Shape (K, K, D)	F	S
EfficientNet_B7	DC	(3, 3, 1)	25024	9
	DC	(5, 5, 1)	45216	25
	PC	(1, 1, 8)	288	8
	PC	(1, 1, 12)	2016	12
	PC	(1, 1, 16)	64	16
	PC	(1, 1, 20)	3360	20
	PC	(1, 1, 32)	312	32
	PC	(1, 1, 40)	9600	40
	PC	(1, 1, 48)	2016	48
	PC	(1, 1, 56)	13440	56
	PC	(1, 1, 64)	48	64
	PC	(1, 1, 80)	3360	80
	PC	(1, 1, 96)	29952	96
	PC	(1, 1, 160)	21120	160
	PC	(1, 1, 192)	56	192
	PC	(1, 1, 224)	13440	224
	PC	(1, 1, 288)	452	288
	PC	(1, 1, 384)	29952	384
	PC	(1, 1, 480)	780	480
	PC	(1, 1, 640)	14080	640
	PC	(1, 1, 960)	2064	960
	PC	(1, 1, 1344)	2960	1344
	PC	(1, 1, 2304)	6496	2304
	PC	(1, 1, 3840)	2400	3840
SC	(3, 3, 3)	64	27	
FC	(2560, 1, 1)	1	2560	

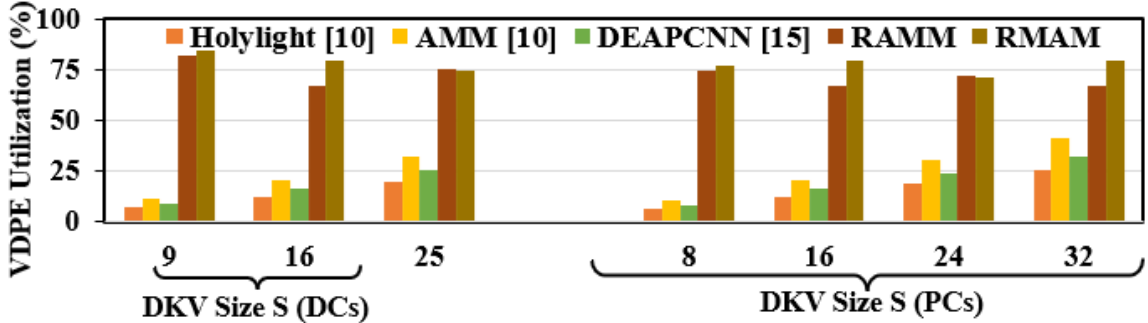


Figure 4.6: VDPE utilization % (utilized VDPE area/total area) for MAM (HOLYLIGHT[144], $N=44$), AMM (DEAPCNN[140], $N=31$), RAMM ($N=31$), and RMAM ($N=43$) at $DR=1GS/s$ and 4-bit precision for various DKV sizes corresponding to DCs and PCs.

4.5 MRR-based Reconfigurable TPC Architectures

In this section, we present a novel reconfigurable VDPE which serves as the backbone of our proposed MRR-based reconfigurable TPC architectures. This reconfigurable VDPE adds the following two desirable attributes to the AMM and MAM types of TPCs. *First*, it introduces the flexibility to the TPCs so that the processing of the DKVs of various sizes can be efficiently mapped onto them, regardless of the fixed VDPE size N for the TPCs. *Second*, it introduces opportunities for increasing the processing parallelism and MRR utilization efficiency. This reconfigurable VDPE can directly replace the VDPEs of the MAM and AMM TPCs (Fig. 4.3), to convert them into reconfigurable MAM (we refer to it as RMAM, henceforth) and reconfigurable AMM (we refer to it as RAMM, henceforth) TPCs. The structure and operation of our invented reconfigurable VDPE are discussed in the following subsections.

4.5.1 Reconfigurable VDPE: Structure and Layout

Fig. 4.7 illustrates our proposed reconfigurable VDPE. It consists of a DKV element, which is an array of N modulation MRRs that can imprint N pointwise products onto the incoming N wavelength channels, similar to the DKV elements of the MAM and AMM TPCs from Fig. 4.3. This DKV element is followed by a group of a total of y pairs of MRR comb switches (CSs). These y CS pairs can re-aggregate the incoming N pointwise-product-modulated wavelength channels (incoming from the DKV element) into a total of y distinct sets, with each set L (Fig. 4.7) having a total of x distinct wavelength channels. In other words, each CS pair filters a comb (set L ; Fig. 4.7) of x distinct wavelength channels from the incoming N wavelength channels. Each CS pair is able to do this because of its innate spectral response that allows it to be in resonance with x distinct wavelengths simultaneously (more on the design of CSs in Section V.C). Each CS pair then sends its corresponding x wavelength channels to its dedicated summation element (SE), which performs a

signed accumulation of the data carried on the x wavelength channels to produce a VDP result. To enable the signed accumulation, similar to the SEs of the AMM and MAM TPCs (Fig. 4.3), the SEs corresponding to the CS pairs also employ balanced PDs. Thus, the group of y CS pairs per reconfigurable VDPE enables y VDP results of size x each to be produced in parallel.

Note that here x , y , and N are integers, and their relation is given by this equation: $y = N > 2x ? \text{floor}(N/x) : 0$. Henceforth, we refer to x as re-aggregation size. Also note that for the group of CS pairs to produce y in-parallel VDP results, each CS pair in the group has to be switched ON by electro-optically tuning its spectral resonance passbands to align with its corresponding set of x wavelength channels [97]. In contrast, it is also possible to switch OFF a CS pair by tuning its resonance passbands out of alignment with the corresponding x wavelength channels. When the CS pairs are switched OFF, all of the N incoming wavelength channels are allowed to pass by the CS pairs to be eventually accumulated at the summation element SE^N . Thus, depending on whether the CS pairs are ON or OFF, the reconfigurable VDPE can operate in two different modes. These modes are further explained in the next section.

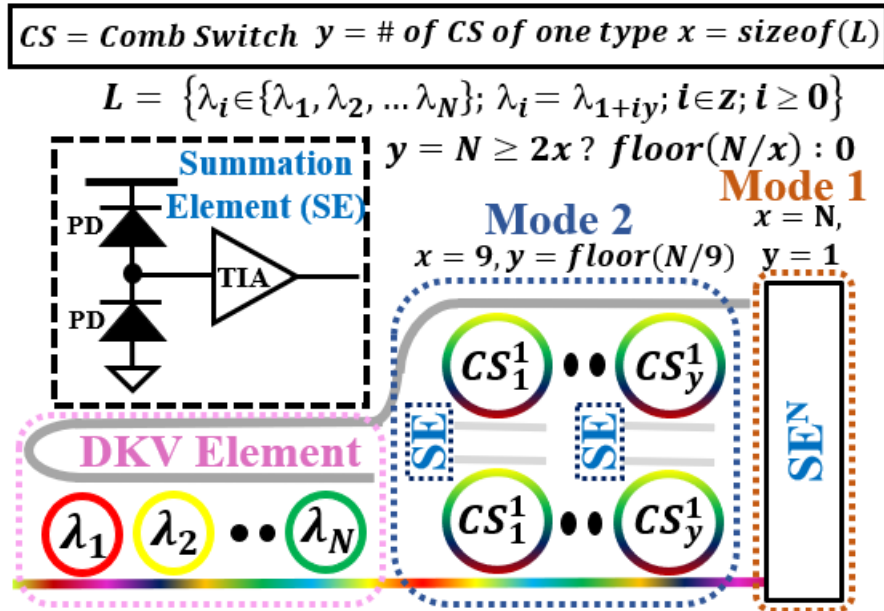


Figure 4.7: Schematic of our invented reconfigurable VDPE, employing a DKV element and one group of comb switch (CS) pairs corresponding to the reconfiguration Mode 2.

4.5.2 Reconfigurable VDPE: Operation

The reconfigurable VDPEs of our RMAM and RAMM TPC architectures support two operational modes, i.e., Mode 1 and Mode 2 (Fig. 4.7). Mode 1 is the non-reconfiguration mode, in which all the CS pairs of a reconfigurable VDPE are switched

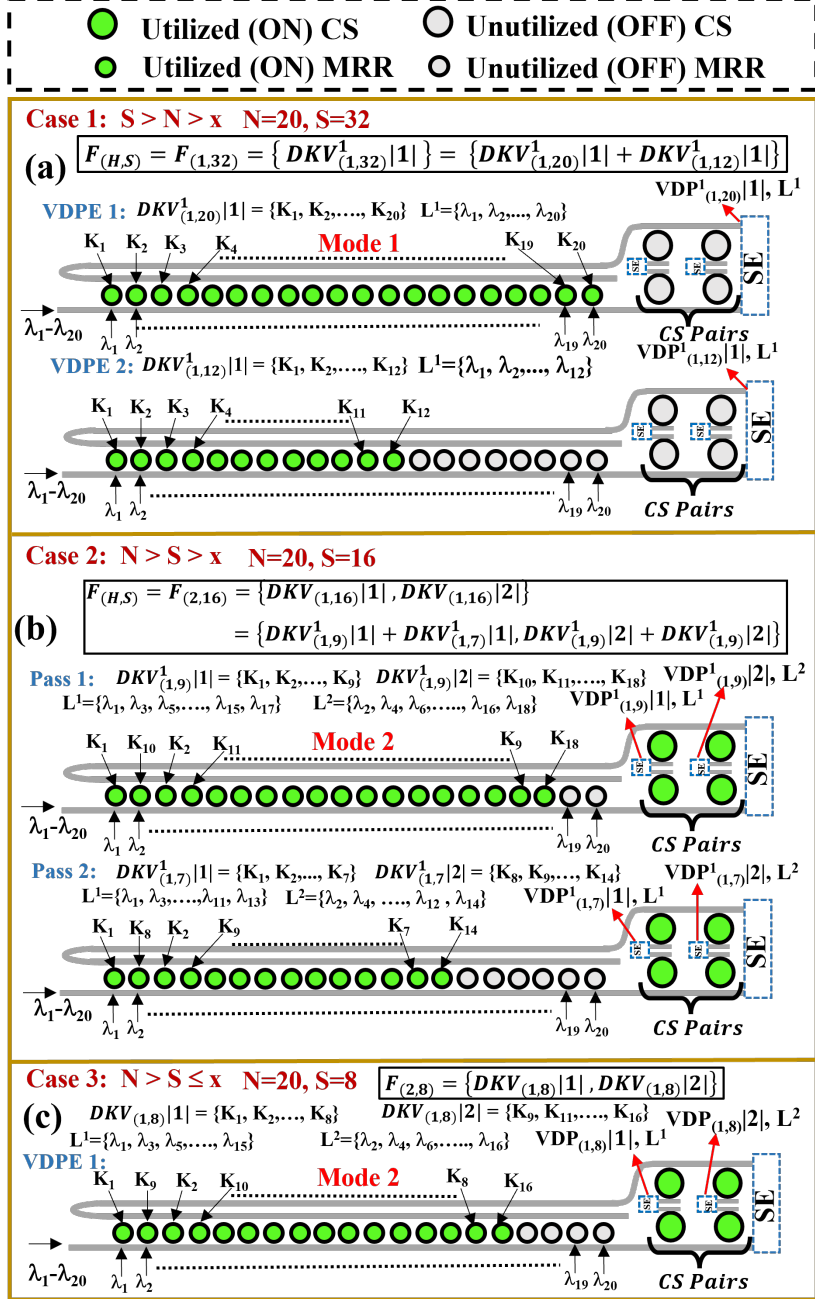


Figure 4.8: Example operation of our reconfigurable VDPE for various cases depending on the S and N values, for $x=9$. Here, ON and OFF CS pairs, respectively, represent Mode 2 and Mode 1 of operation.

OFF, so that the reconfigurable VDPE operates like a regular VDPE to produce a VDP result of size N . In contrast, Mode 2 is the reconfiguration mode, in which all the CS pairs of a reconfigurable VDPE are switched ON so that a total of y in-parallel VDP results can be produced with each result being of size x . We determine the re-aggregation size x to be 9 because the DKV size $s=9$ is the most common,

frequently used, smallest DKV size across various CNNs (Table 4.3). We reason that determining the value of x based on the most common, smallest DKV size maximizes the opportunities for increasing processing parallelism and utilization efficiency.

Since the reconfigurable VDPEs simply replace the regular VDPEs of the MAM and AMM TPCs, an RAMM/RMAM TPC is analogous in structure to an AMM/MAM TPC. From Fig. 4.3 in Section III, an AMM/MAM TPC contains a VDPE block (containing multiple VDPEs), which is basically the structure that remains in the TPC if we mask off the DIV element of the TPC. Similarly, an RAMM/RMAM TPC would also typically contain a block of M reconfigurable VDPEs. Since each reconfigurable VDPE is of size N , such reconfigurable VDPE block would basically be of $M \times N$ dimensions.

From Section II and III, the input kernel tensors are flattened into DKVs of size S and then mapped onto the VDPE block of a TPC for processing. In that vein, to further explain our method of mapping for RAMM/RMAM TPCs, typically, a matrix $F_{(H,S)}$, which has H rows with each row containing a DKV of size S , is mapped onto one or multiple $M \times N$ sized reconfigurable VDPE blocks. This matrix $F_{(H,S)}$ can be written as a set of a total of $H \times 1 \times S$ sized DKVs, i.e., $F_{(H,S)} = \{DKV_{(1,S)}|1|, \{DKV_{(1,S)}|2|, \dots, DKV_{(1,S)}|H|\}$. Each individual DKV in this set is basically a flattened kernel tensor. When matrix $F_{(H,S)}$ is mapped onto reconfigurable VDPE blocks, this set of DKVs is mapped onto the individual reconfigurable VDPEs of the blocks. After this mapping, the individual reconfigurable VDPEs operate in either Mode 1 or Mode 2, depending on how the size S of the DKVs compares with the size N of the reconfigurable VDPEs, given that the VDPEs have the re-aggregation size $x=9$.

We advocate for selecting the most appropriate mapping and mode of operation (from Mode 1 or Mode 2) that can maximize the MRR utilization and processing throughput of the RAMM/RMAM TPCs. We identify three cases for the relation among N , S , and $x=9$ that drive the selection of the appropriate mapping and mode of operation. The mappings and modes of operation for these three cases are illustrated in Fig. 4.8. For Fig. 4.8, we selected the example RAMM TPC architecture from Table 4.2 with $N=20$ for DR = 3 GS/s. These illustrations are further explained below.

Case 1, $S > N > x$: For this case, the reconfigurable VDPEs operate in Mode 1. In this case, before mapping the DKV matrix $F_{(H,S)}$ on the reconfigurable VDPEs, matrix $F_{(H,S)}$ is divided into multiple slices along the dimension S . Since S can be written as $S=b \times N+c$, the matrix $F_{(H,S)}$ is divided into a total of $b+1$ slices, with b slices of size (H, N) each and one slice of size (H, c) . Hence, $F_{(H,S)}$ can be written as $\{F_{(H,N)}^1 + \dots + F_{(H,N)}^b + F_{(H,c)}^1\}$, where $F_{(H,N)}^1, \dots, F_{(H,N)}^b$, and $F_{(H,c)}^1$ are the slices of matrix $F_{(H,S)}$. Here, '+' represents the concatenation operator. Since matrix $F_{(H,S)}$ can be written as $\{DKV_{(1,S)}|1|, \dots, DKV_{(1,S)}|H|\}$ (as discussed earlier), slicing of $F_{(H,S)}$ in turn means slicing of all the DKV components of $F_{(H,S)}$. Consequently, every component DKV $DKV_{(1,S)}|H|$ of $F_{(H,S)}$ is also divided into $b+1$ slices along the dimension S . Hence, $DKV_{(1,S)}|H|$ can be written as $\{DKV_{(1,N)}^1|H| + \dots + DKV_{(1,N)}^b|H| + DKV_{(1,c)}^1|H|\}$. Similarly, each DKV of matrix $F_{(H,S)}$ can be re-written

as the concatenation of DKV slices. After the matrix $F_{(H,S)}$ has been sliced, each individual slice of every DKV of $F_{(H,S)}$ is mapped onto one reconfigurable VDPE running in Mode 1 operation. Each VDPE generates a partial VDP result corresponding to the mapped DKV slice. The partial VDP results from all the DKV slices that originated from a single original DKV (a component of $F_{(H,S)}$) are then accumulated at the *psum* reduction network (not shown in the figure), to generate the final VDP result. Case 1 is shown in Fig. 4.8(a), where input matrix $F_{(H,S)}$ with $H=1$ and $S=32$ is sliced based on $b=1$, $c=12$. The consequently generated individual DKV slices are then mapped on to reconfigurable VDPE1 and VDPE2 operating in Mode 1. The two VDPEs produce two partial VDP results $VDP_{(1,20)}^1|1|$ and $VDP_{(1,12)}^1|1|$. These partial VDP results are then summed together.

Case 2, $N > S > x$: For this case, the reconfigurable VDPEs operate in Mode 2. In this case, before mapping the DKV matrix $F_{(H,S)}$ onto the reconfigurable VDPEs, matrix $F_{(H,S)}$ is divided into multiple slices along the dimension S . Since S can be written as $S=b \times x + c$ (the values b and c here are often different from Case 1), the matrix $F_{(H,S)}$ is divided into a total of $b+1$ slices, with b slices of size (H, x) each and one slice of size (H, c) . Hence, $F_{(H,S)}$ can be written as $\{F_{(H,x)}^1 + \dots + F_{(H,x)}^b + F_{(H,c)}^1\}$, where $F_{(H,x)}^1, \dots, F_{(H,x)}^b$, and $F_{(H,c)}^1$ are the slices of matrix $F_{(H,S)}$. Here, '+' represents the concatenation operator. Since matrix $F_{(H,S)}$ can be written as $\{DKV_{(1,S)}|1|, \dots, DKV_{(1,S)}|H|\}$ (as discussed earlier), slicing of $F_{(H,S)}$ in turn means slicing of all the DKV components of $F_{(H,S)}$. Consequently, every component $DKV_{(1,S)}|H|$ of $F_{(H,S)}$ is also divided into $b+1$ slices along the dimension S . Hence, $DKV_{(1,S)}|H|$ can be re-written as $\{DKV_{(1,x)}^1|H| + \dots + DKV_{(1,x)}^b|H| + DKV_{(1,c)}^1|H|\}$. After the matrix $F_{(H,S)}$ has been sliced, the total of H DKV slices that belong to every matrix slice $F_{(H,x)}^1$ are mapped onto a total of (H/y) different reconfigurable VDPEs, where each reconfigurable VDPE processes a total of y slices ($y = N \geq 2x ? \text{floor}(N/9) : 0$). Hence, each reconfigurable VDPE in this case produces a total of y VDP results in parallel, which basically renders a *y* times throughput improvement for each reconfigurable VDPE, compared to Mode 1 of operation. In this case, once a given matrix slice has been mapped onto the total of (H/y) VDPEs, the partial VDP results are produced for all DIVs from the input CNN layer that correspond to the mapped matrix slice in a stationary weight dataflow, before the next matrix slice is mapped for a new pass of all the DIVs. The partial VDP results from all the DKV slices that originated from a single original DKV (a component of $F_{(H,S)}$) are then accumulated at the *psum* reduction network, to generate the final VDP result. Case 2 is shown in Fig. 4.8(b), where input matrix $F_{(H,S)}$ with $H=2$ and $S=16$ is sliced based on $b=1$, $c=7$. The consequently generated individual DKV slices are then mapped on to a single VDPE operating in Mode 2 in two passes. Pass 1 generates $VDP_{(1,9)}^1|1|$ and $VDP_{(1,9)}^1|2|$, Whereas Pass 2 generates $VDP_{(1,7)}^1|1|$ and $VDP_{(1,7)}^1|2|$. The partial VDP results from Pass 1 are then summed together with respective partial VDP results from Pass 2 to generate two final VDP results.

Case 3, $N > S \leq x$: For this case, the reconfigurable VDPEs operate in Mode 2. The input matrix $F_{(H,S)}$ is directly mapped onto the individual reconfiguration VDPEs, after writing it in terms of DKVs as $F_{(H,S)} = \{DKV_{(1,S)}|1|, \dots, DKV_{(1,S)}|H|\}$.

Here, H DKVs corresponding to matrix $F_{(H,S)}$ are mapped onto a total of (H/y) reconfigurable VDPEs operating in Mode 2, where each VDPE processes a total of y DKVs in parallel. Case 3 is shown in Fig. 4.8(c), where input matrix $F_{(H,S)}$ with $H=2$ and $S=8$ is mapped on to a reconfigurable VDPE operating in Mode 2. The VDPE processes $y=2$ DKVs, generating $y=2$ VDP final results $VDP_{(1,8)}|1|$ and $VDP_{(1,8)}|2|$.

Discussion: Compared to Mode 1, Mode 2 operation of the reconfigurable VDPE increases the hardware/MRR utilization, while simultaneously increasing the processing throughput by up to $y\times$. This throughput improvement makes it tolerable to have the extra area overhead of additional y CS pairs per reconfigurable VDPE for Mode 2 operation. It can be argued that $y\times$ improvement in throughput can also be achieved by employing $y\times$ more VDPEs (without CS pairs) in Mode 1 operation. However, employing $y\times$ more VDPEs in Mode 1 operation incurs the area overhead of additional $N\times y$ MRRs (as each VDPE has N MRRs), whereas employing y CS pairs in one reconfigurable VDPE incurs the area overhead equivalent to additional $6\times y$ MRRs (as the area of 1 CS pair = area of 6 MRRs). Since for all TPCs from Table 4.2, $6\times y$ is less than $N\times y$, Mode 2 operation turns out to be highly efficient than Mode 1 operation. As for modern CNNs, more than 40% of the DKVs (Table 4.3) belong to Case 2 ($N>S>x$) and Case 3 ($N>S\leq x$) above, the substantially improved efficiency for Mode 2 operation also translates in up to 78.2% and 54.71% higher VDPE utilization, respectively, for our RAMM and RMAM TPCs in Fig. 4.6, compared to their respective baseline AMM (DEAPCNN) and MAM (HOLYLIGHT) TPCs.

4.5.3 Design of MRR Comb Switches

From Section V.A, a CS has the capability of filtering a comb of x wavelength channels from the incoming N wavelength channels [97]. Generally, the resonance passband of a modulation MRR periodically repeats at the spectral distance known as Free Spectral Range (FSR). Therefore, to filter x distinct wavelength channels simultaneously, the FSR of the CS needs to match with the inter-channel spacing so that the periodic passbands of the CS overlap with the x wavelength channels. Since N and inter-channel spacing vary for our RAMM and RMAM TPCs across different DRs (Table 4.2), the required FSR for the CSs would also vary across different DRs. The FSR for a CS can be defined by appropriately defining its radius [97]. Therefore, we designed the CSs with desired FSRs required for our RAMM and RMAM TPCs for various DRs by appropriately defining the radius of a primitive MRR. For that, we used the photonics foundry-validated MODE and INTERCONNECT tools from Ansys/Lumerical [72]. Table 4.4 lists the design parameters of our designed CSs. From Table 4.4, a CS can incur insertion losses. This can impact the achievable N for a TPC. Our scalability analysis in Section 4.3.2 accounts for this fact.

4.5.4 System Level Implementation

Fig. 4.9 illustrates the system level implementation of our accelerators. It consists of a global memory for storing CNN parameters, a pre-processing and mapping unit for decomposing the tensors into DIVs/DKVs and mapping them onto the DIV/DKV

elements. It has a mesh of tiles connected to routers and this mesh network facilitates parameter communication among tiles. Each tile consists of 4 RMAM/RAMM TPCs interconnected (via H-tree network) with output buffer, activation and pooling units. Due to their analog nature, the RMAM/RAMM TPCs require DACs and ADCs as well. In addition, each tile also contains *psum* reduction network to enable summing up of the intermediate *psums*. Depending on the type of convolution operations being processed (SC/DC/PC) and their tensor sizes (Table 4.3), the mapping unit sends control signals to the individual RAMM/RMAM TPCs to reconfigure their operational modes (Mode 1/Mode 2).

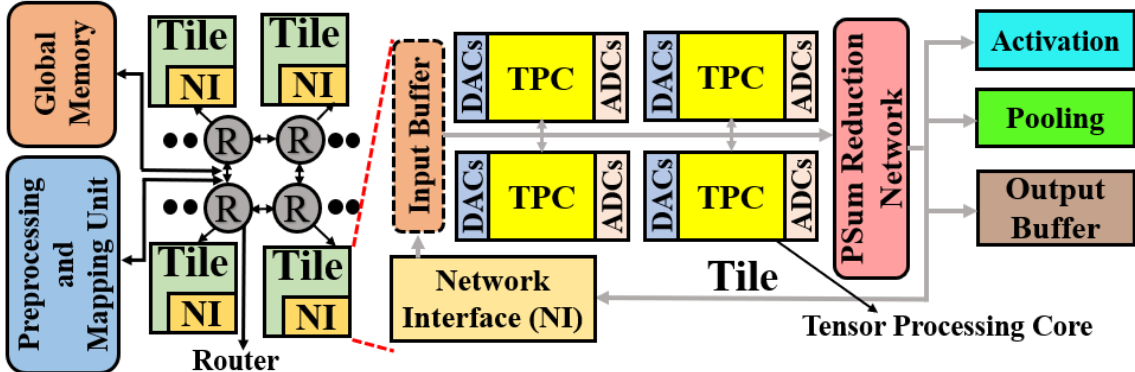


Figure 4.9: System level overview of a CNN accelerator that employs our RMAM/RAMM TPCs.

4.6 Evaluation

4.6.1 Simulation Setup

To evaluate our designed RAMM and RMAM accelerator architectures, we simulated the inference of various depthwise separable convolutions based CNNs such as EfficientNetB7 [170], Xception [36], NASNetMobile[216], and ShuffleNetV2 [208] with input batch size of 1. These CNNs comprehensively cover wide variations seen in CNNs in terms of tensor sizes. We developed a custom, transaction-level, cycle-true python-based simulator to model CNN inference on MRR-based TPC accelerators with weight-stationary dataflow.

We compared our RAMM and RMAM accelerator architectures with the baseline AMM (DEAPCNN [140]), MAM (HOLYLIGHT [144]) and the latest variant of AMM design (CROSSLIGHT [116]). We evaluate these accelerators at 4-bit precision and across different DRs such as 1 GS/s, 3 GS/s, and 5 GS/s. The N values corresponding to different DRs are taken from Table 4.2 for our system level analysis. Table 4.5 and Table 4.6 give the parameters used for evaluating the overheads of the peripherals. We consider each laser diode to emit input optical power of 10 mW (10 dBm) (Table 4.1) [140]. Multiplexer and splitter parameters are taken from [144], and other VDP element parameters are listed in Table 4.7.

We performed area proportionate (AP) analysis, for which, we altered the VDPE count of each accelerator so that the accelerators’ area matched with the area of the RMAM accelerator with VDPE count of 512. Table 4.8 reports our obtained area-proportionate VDPE counts for all our considered accelerators. We evaluate the metrics such as Frames Per Second (FPS) and FPS/W (energy efficiency) for our considered accelerator architectures.

4.6.2 Evaluation Results

Fig.4.10 shows the FPS results for various accelerators at different DRs, normalized to RMAM at 1 GS/s. Our RMAM accelerator on gmean outperforms MAM (HOLYLIGHT), AMM (DEAPCNN), and CROSSLIGHT for all DRs, such as 1 GS/s, 3 GS/s, and 5 GS/s. At 1 GS/s, RMAM achieves 1.8×, 17.1×, and 65× better FPS than MAM (HOLYLIGHT), AMM (DEAPCNN), and CROSSLIGHT, respectively, on gmean across the CNNs. From Section 4.3.2, the N values decrease with increase in DR, which leads to lower throughput with increase in DR for various accelerators. For instance, RMAM’s FPS drops by 5.3× and 8× at 3 GS/s and 5 GS/s respectively, compared to its FPS at 1 GS/s. Therefore, compared to the 3-GS/s (5-GS/s) variants of MAM (HOLYLIGHT), AMM (DEAPCNN), and CROSSLIGHT, respectively, our 1-GS/s RMAM variant achieves 8.3× (10.2×), 52.57× (79.8×), and 86× (106×) better FPS. These FPS benefits are because of our RMAM variants’ higher throughput and more efficient processing through reconfiguration (Mode 2 operation). Our other accelerator RAMM, at 1 GS/s, achieves 1.54× and 5.8× better FPS compared to AMM (DEAPCNN) and CROSSLIGHT, respectively. Even at higher DRs, RAMM performs better than AMM (DEAPCNN) and CROSSLIGHT. However, at 5 GS/s, because of low N , reconfiguration is not supported in RAMM due to the condition $y = N \geq 2x? \text{floor}(N/9) : 0$. Therefore RAMM is exactly identical to AMM (DEAPCNN) at 5 GS/s. Overall, our RMAM accelerator gives better FPS compared to other accelerators across all DRs.

Fig. 4.11 shows the FPS/W (energy efficiency) results for various accelerators across different DRs, normalized to RMAM at 1 GS/s. Our RMAM and RAMM accelerators also achieve better FPS/W compared to the other accelerators. At 1 GS/s, RMAM achieves 1.5×, 27.2×, and 171× better FPS/W than MAM (HOLYLIGHT), AMM (DEAPCNN), and CROSSLIGHT, respectively, on gmean across various CNNs. Similar to the FPS results, when compared to the 3-GS/s (5-GS/s) variants of MAM (HOLYLIGHT), AMM (DEAPCNN), and CROSSLIGHT, our 1-GS/s RMAM variant achieves 4.2× (4×), 46.4× (29.6×), and 80.7× (54×) better FPS/W, respectively. These energy efficiency benefits are also because of the improved VDPE utilization achieved from the superior reconfigurability of our RMAM accelerator. The improved VDPE utilization better amortizes the static power consumption of the constituent hardware components (e.g., MRRs, CSs, ADCs, DACs) to yield better energy efficiency. Our other accelerator RAMM, at 1 GS/s, achieves 1.5× and 9.7× better FPS/W compared to AMM (DEAPCNN) and CROSSLIGHT, respectively. Even at higher DRs, RAMM performs better than AMM (DEAPCNN)

and CROSSLIGHT. However, at 5 GS/s, RAMM and AMM (DEAPCNN) have equal energy efficiency. Overall, our RMAM provides better energy efficiency compared to the other accelerators across different DRs.

4.7 Summary

In this Chapter, we presented the use of our invented reconfigurable VDPEs as a novel way of introducing flexibility in the photonic MRR-based CNN accelerators. Our reconfigurable VDPEs employ a set of comb switches to enable dynamic maximization of the size compatibility between the VDPEs and the CNN tensors that are processed using the VDPEs. We then used our reconfigurable VDPEs to enhance the MRR-based CNN accelerators of the AMM and MAM categories. Consequently, we derived different variants of Reconfigurable MAM (RMAM) and Reconfigurable AMM (RAMM) accelerators that operate at different data rates. We evaluated different variants of our RMAM and RAMM accelerators against three prior works, for the inference of four modern CNNs with mixed-sized tensors. Our evaluation indicates that our RMAM and RAMM accelerators are significantly better at striking a balance between the hardware utilization and CNN processing latency, which in turn provides them with substantial improvements in FPS and FPS/W, with equal area consumption, compared to the photonic MRR-based accelerators from prior works. These results promote the use of our RMAM and RAMM accelerators for efficient processing of future CNNs having a wide variety in their employed tensor sizes.

Acknowledgments

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

Table 4.4: Design parameters of various comb switch (CS) designs used in our RMAM and RAMM TPCs for various DRs.

Data Rate (DR) (GS/s)	1	3	5
RAMM TPC			
N	31	20	16
CS_{FSR}	4.83nm	5 nm	NA
Radius	18.17 μm	17.5 μm	NA
No of CS Pairs	3	2	0
Insertion Loss (dB)	0.029	0.028	0
RMAM TPC			
N	43	28	22
CS_{FSR}	4.65 nm	5.35nm	4.54 nm
Radius	18.98 μm	16.2 μm	19.49 μm
No of CS Pairs	4	3	2
Insertion Loss (dB)	0.029	0.026	0.031

Table 4.5: ADC area and power overheads.

ADC	Area (mm^2)	Power
1 GS/s [122]	0.002	2.55 mw
3 GS/s [147]	0.021	11mw
5 GS/s [60]	0.103	29 mw

Table 4.6: Accelerator Peripherals Parameters [144].

	Power(mW)	Area(mm^2)	Latency
DAC [82]	30	0.034	0.78ns
Reduction Network	0.05	0.03E-3	3.125ns
Activation Unit	0.52	0.6E-3	0.78ns
IO Interface	140.18	24.4E-3	0.78ns
Pooling Unit	0.4	0.24E-3	3.125ns
eDRAM	41.1	166E-3	1.56ns
Bus	7	9E-3	5 cycles
Router	42	0.151	2 cycles

Table 4.7: VDP Element Parameters [116].

DKV/DIV MRR Q-factor	8000	
DKV/DIV MRR FWHM	0.2 nm	
Sensitivity of PD	-20 dBm	
	Power (mW)	Latency
EO Tuning	80 $\mu\text{W}/\text{FSR}$	20 ns
TO Tuning	27.5 mW/FSR	4 μs
TIA	7.2 mW	0.15 μs
Photodetector	2.8 mW	5.8 ps

Table 4.8: VDPE counts of various accelerators.

Accelerators	DR (GS/s)		
	1	3	5
RMAM	512	512	512
RAMM	587	576	567
MAM (HOLYLIGHT [144])	568	562	547
AMM (DEAPCNN [140])	656	629	620

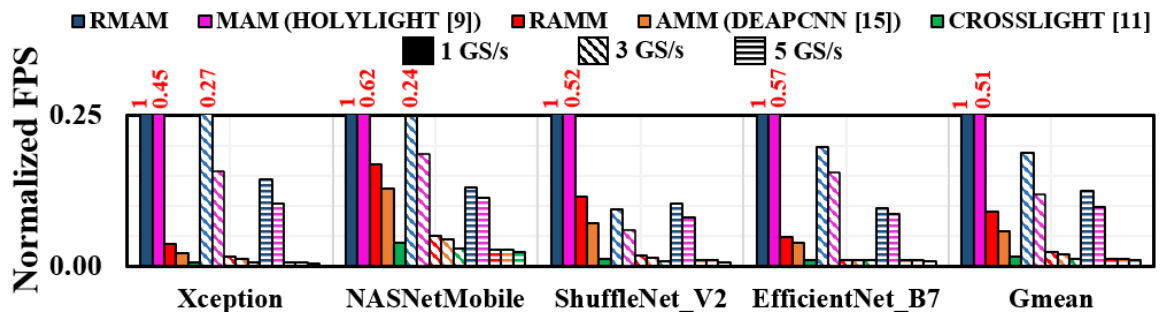


Figure 4.10: Area proportionate comparison of FPS for various accelerators across different CNNs and data rates (DRs). Results are normalized with respect to RMAM at 1 GS/s.

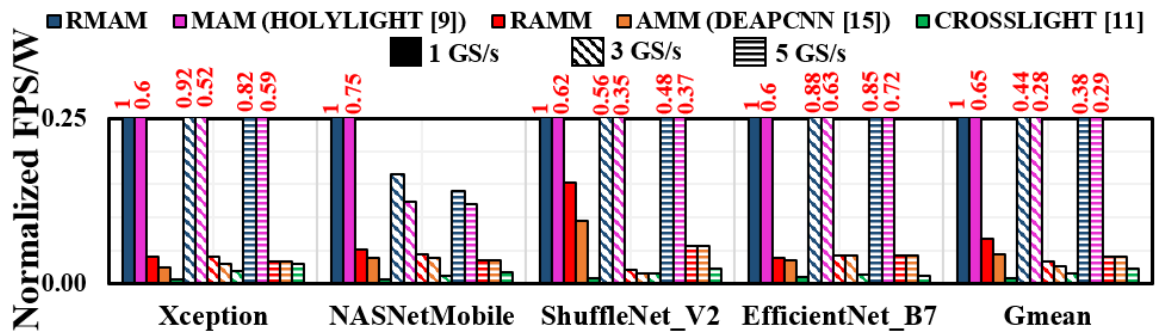


Figure 4.11: Area proportionate comparison of FPS/W for various accelerators across different CNNs and data rates (DRs). Results are normalized with respect to RMAM at 1 GS/s.

Chapter 5 An Optical XNOR-Bitcount Based Accelerator for Efficient Inference of Binary Neural Networks

5.1 Introduction

Convolutional Neural Networks (CNNs) have revolutionized the implementation of various artificial intelligence tasks, such as image recognition, language translation, and autonomous driving [96, 47], 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 [61, 214, 56]. 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 [132]. 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-photonics accelerators have shown great promise to provide unparalleled parallelism, ultra-low latency, and high energy efficiency [215, 160]. Prior work [215] utilizes microdisks to realize XNOR-Bitcount processing cores (XPCs) that process the input and weight vectors, whereas [160] 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 (*psums*). Accumulating such a large number of *psums* to obtain the final result, using a *psum* reduction network, can incur a very high latency overhead.

To address these shortcomings, this Chapter 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 *psums*, 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 Chapter 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).

5.2 Preliminaries

5.2.1 Binary Neural Networks (BNNs)

BNNs are specific types of CNNs that employ quantization techniques [205] 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 [40, 132]. In general, the *sign* function is the most widely used binary quantization function (Q):

$$Q(x) = \text{sign}(x) = x \geq 0 ? +1 : -1 \quad (5.1)$$

Like for CNNs [22], 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. 5.2.

$$z = W \odot I = \sum_{i=1}^S W_i \odot I_i \quad (5.2)$$

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 $\text{sign}(z)$ 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 bitcount result z , where z_{max} is the size of the binarized vectors I and W .

5.2.2 Processing of BNNs on Hardware

Fig. 5.1(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. 5.1(a) with red, blue, yellow, and green borders), generating one output value per input channel window. From Fig. 5.1(b), to perform one such inner product (i.e., corresponding to the input channel window highlighted in green in Fig 5.1(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. 5.1(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. 5.1(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.

5.2.3 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., [109, 157, 206]). These accelerators can be classified as incoherent (e.g., [109, 157, 22]) or coherent (e.g., [44, 107]). Because of the inherent advantages of incoherent accelerators [157][152], the BNN-specific incoherent accelerators [160] and [215] were reported. *These optical BNN accelerators from prior works employ binary value set $\{0, 1\}$.* [160] proposes broadcast and weight styled [34] XNOR-Bitcount circuits, which use heterogeneous MRRs to mitigate fabrication process variations. In contrast, the microdisk-based accelerator [215] 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 [160] and [215] require at least two MRRs or microdisks to perform a 1-bit XNOR operation (in [215], 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 $psums$ temporarily in

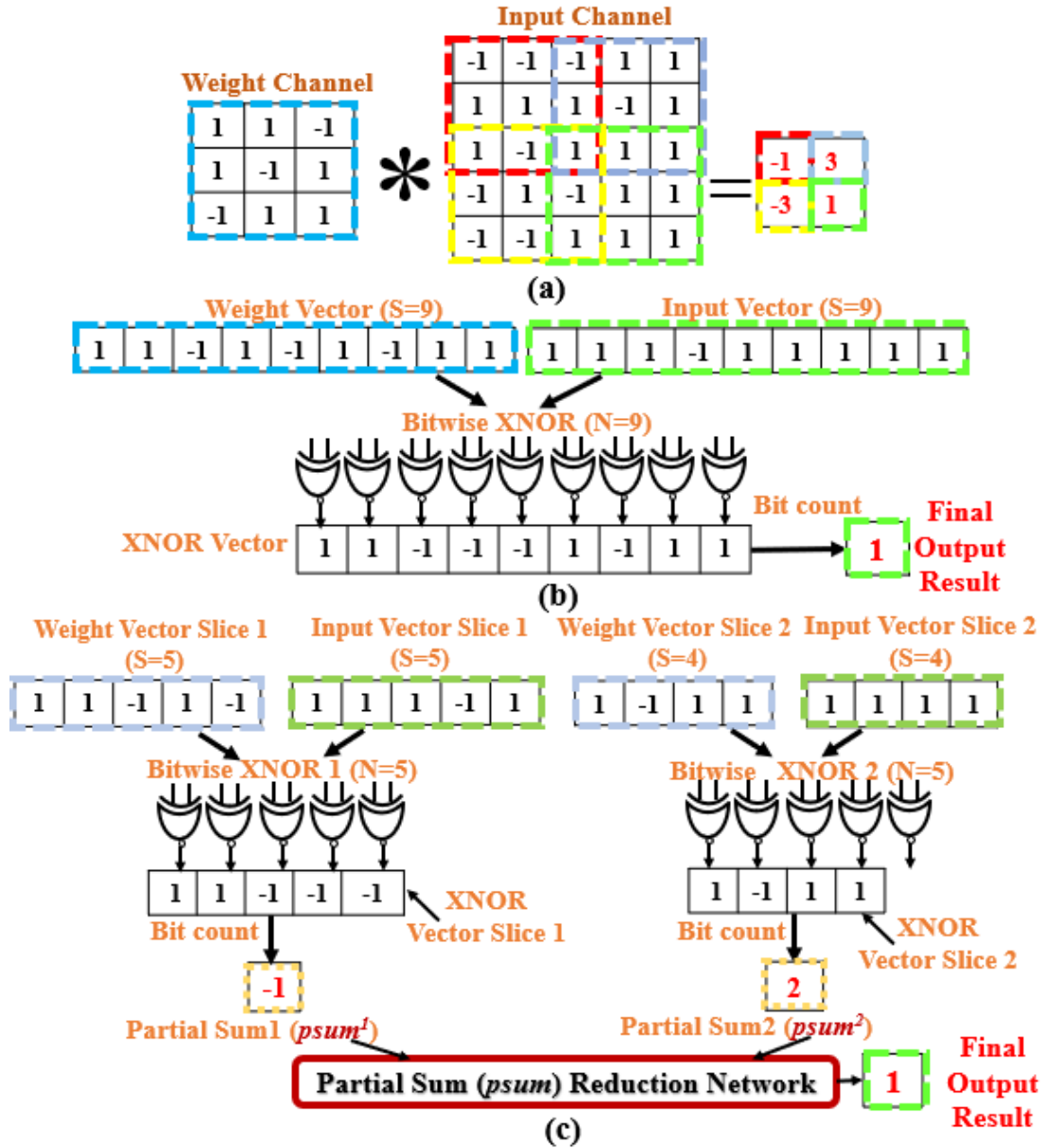


Figure 5.1: (a) Illustration of a convolution between a weight and input channel in a Binary Neural Network. Bit-wise XNOR and bitcount operations between a flattened weight vector and input vector, (b) when $S=N=9$, and (c) when $N=5$, $S=9$; each input and weight vector of $S=9$ is split into two slices (Slice 1 with $S=5$ and Slice 2 with $S=4$). Binary value set $\{-1,1\}$ is used in this example.

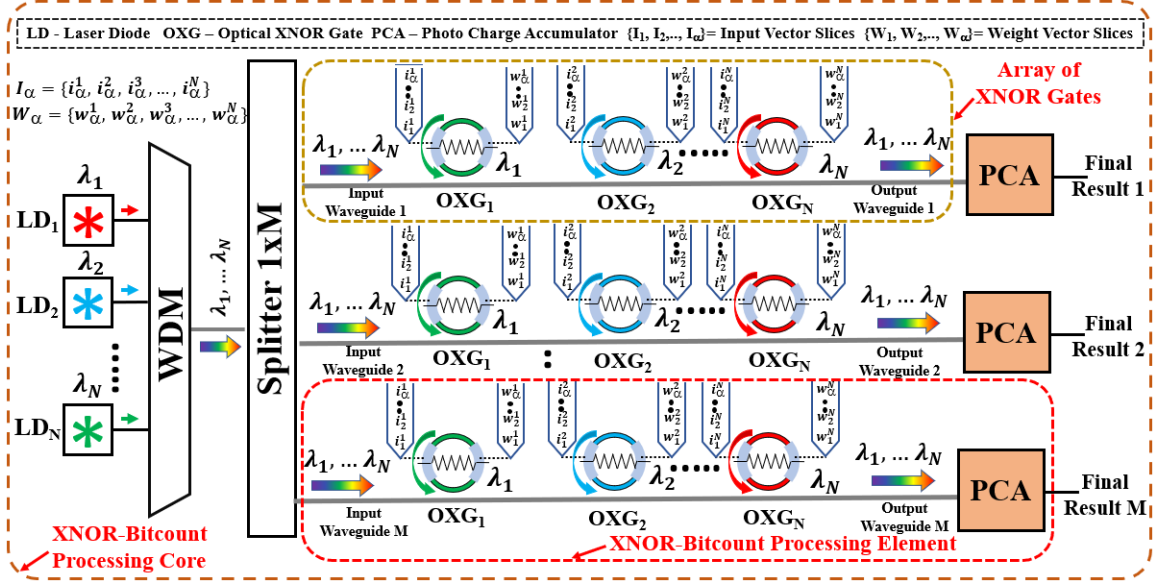


Figure 5.2: Schematic of an XNOR-Bitcount Processing Core (XPC) of our OXBNN accelerator. Our OXBNN employs binary value set $\{0,1\}$.

memory. Once sufficient *psums* 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 *psums*, and high latency and energy for processing *psums*. Our OXBNN accelerator addresses these shortcomings of prior works.

5.3 Our Proposed OXBNN Architecture

5.3.1 Overview

The main processing unit of our OXBNN architecture is an XNOR-Bitcount Processing Core (XPC), which is illustrated in Fig. 5.2. 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. 5.2). An XPC contains a total of M XPEs.

5.3.2 XNOR-Bitcount Processing Element (XPE)

From Fig. 5.2, 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.

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. 5.2. 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 $I_1 = \{i_1^1, i_1^2, \dots, i_1^N\}$ and an N -bit weight vector slice $W_1 = \{w_1^1, w_1^2, \dots, w_1^N\}$ 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. 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 $\{I_1, I_2, \dots, I_\alpha\}$ and $\{W_1, W_2, \dots, W_\alpha\}$ 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. 5.3(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. 5.3(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. 5.3(a)), to consequently tune its operand-independent resonance from its fabrication-defined initial position η to its programmed position κ (blue passband; Fig. 5.3(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. 5.3(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

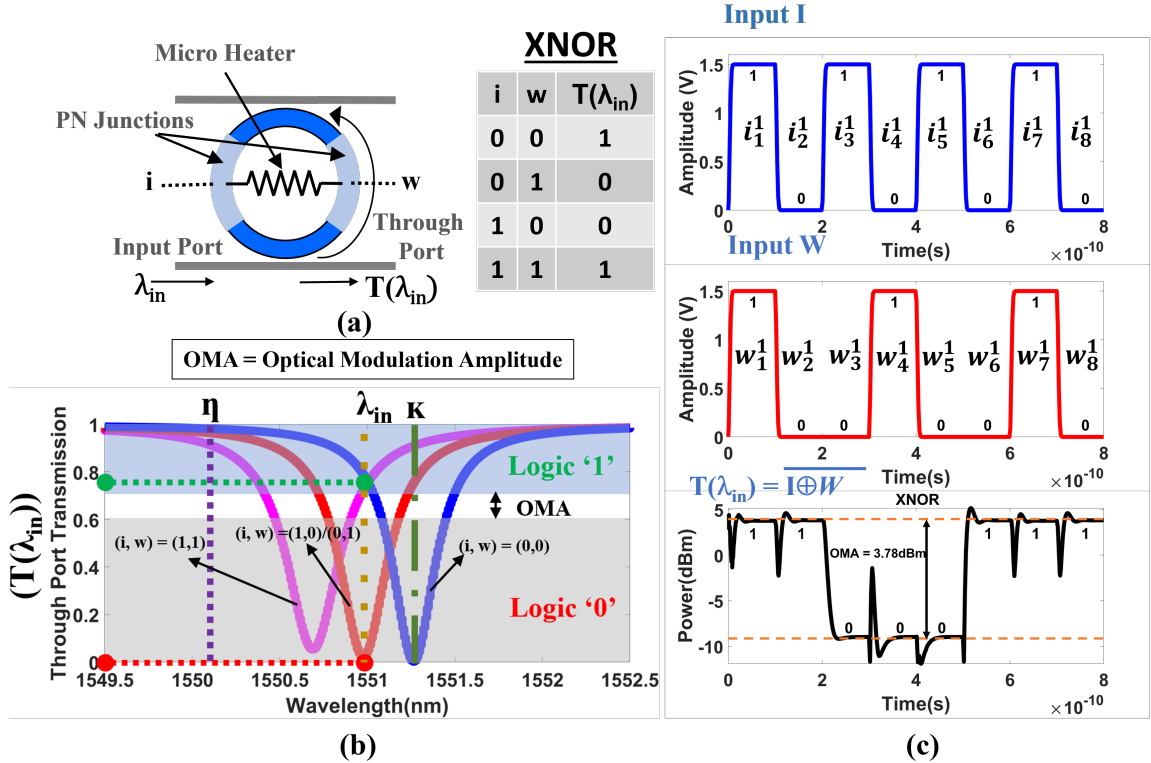


Figure 5.3: (a) Schematic of our Optical XNOR Gate (OXG). (b) Spectral operation of OXG. (c) Transient analysis of OXG.

shown in Fig. 5.3(c). For that, we modelled and simulated our OXG using the foundry-validated tools from Ansys/Lumerical's DEVICE, CHARGE, and INTERCONNECT suites [6]. Fig. 5.3(c) shows two input bit-streams $I = \{i_1^1, i_2^1, \dots, i_8^1\}$ and $W = \{w_1^1, w_2^1, \dots, w_8^1\}$ 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. 5.3 (c), we can say $T(\lambda_{in}) = \{i_1^1 \odot w_1^1, \dots, i_8^1 \odot w_8^1\}$, 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 0.011mm².

Photo-Charge Accumulator (PCA)

From Section 5.3.1, 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 [150] (one of the TIR1 and TIR2 circuits remains redundant, enabled by the demux and mux; Fig 5.4). 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'

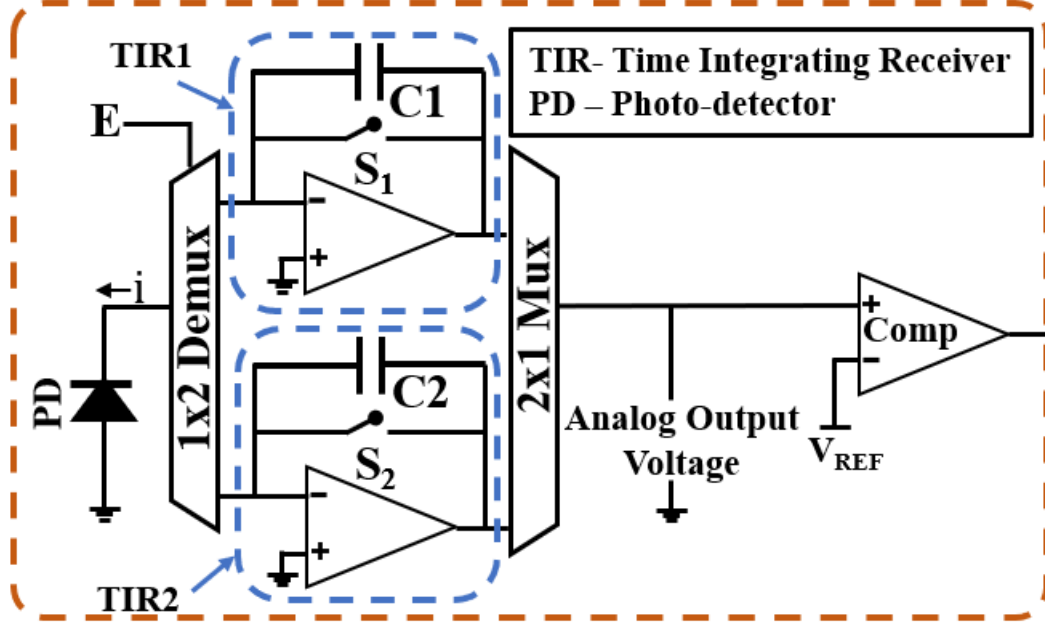


Figure 5.4: Photo-Charge Accumulator (PCA) Circuit. V_{REF} is the threshold required in the *compare()* function discussed in Section 5.2.1. Typically, $V_{REF} = 2.5V$ because we consider the dynamic range of TIR to be 5V.

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 [150]. Hence, when more optical '1's are incident upon the photodiode, 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 [150]. 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. 5.4) 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 5.2.1). 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.

5.4 Scalability Analysis and Mapping

5.4.1 Scalability of XNOR-Bitcount Processing Cores (XPCs)

To determine the achievable size N for our XPC, we adopt scalability analysis equations (Eq. 5.3, Eq. 5.4, and Eq. 5.5) from [9] and [152]. Table 5.1 reports the definitions of the parameters and their values used in these equations. We considered Free Spectral Range (FSR=50nm) [9], FWHM=0.35nm (refer Section 5.3.2), 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 [17, 87, 189], which is accounted for as part of parameter $IL_{penalty}$ in the equations (Table 5.1)). 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. 5.3 and Eq. 5.4 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. 5.4 for N with the obtained set of P_{PD-opt} values across the set of DRs. Table 5.2 reports the achievable N for our XPC across various DRs. 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., $N=66 < (FSR/0.7nm)$).

$$B = \frac{1}{6.02} \left[20 \log_{10} \left(\frac{R_s \times P_{PD-opt}}{\beta \sqrt{\frac{DR}{\sqrt{2}}}} - 1.76 \right) \right] \quad (5.3)$$

$$\beta = \sqrt{2q(R_s P_{PD-opt} + I_d) + \frac{4kT}{R_L} + R_s^2 P_{PD-opt}^2 R I N} \quad (5.4)$$

$$P_{Laser} = \frac{10^{\frac{\eta_{WG}(dB)[N(d_{OXG})+d_{element}]}{10}} M}{\eta_{SMF} \eta_{EC} \eta_{WPE} IL_{i/p-OXG}} \times \frac{P_{PD-opt}}{IL_{penalty}} \times \frac{1}{(OBL_{OXG})^{N-1} (EL_{splitter})^{\log_2 M}} \quad (5.5)$$

Analysis of PCA's Accumulation Capacity: We modeled the photodetector (PD) of our PCA circuit using the INTERCONNECT tool from Ansys/Lumerical [6] for PD responsivity = 1.2 A/W across different P_{PD-opt} values corresponding to the N values in Table 5.2. 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 [2] based model of the PCA with C1=C2=10pF [150], 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

Table 5.1: Definition and values of various parameters used in Eq. 5.3, Eq. 5.4, and Eq. 5.5 (from [9]) for the scalability analysis. Definitions of PCA parameters γ and α .

Parameter	Definition	Value
$P_{Laser}=P_{\lambda_i}^{in}$	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 5.2
α	PCA's Accumulation Capacity (# of XNOR Vectors Slices)	Table 5.2
γ	PCA's Accumulation Capacity (# of accumulated '1's)	Table 5.2

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 5.2. 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 5.2 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 5.4.3.

5.4.2 Mapping Convolutions on an XPC

As described in Section 5.2.1, for processing a BNN convolution on hardware, both the weight and input channels are flattened into binarized vectors. For mapping of a binary convolution on an XPC (or XPE), these binarized input and weight vec-

Table 5.2: 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

tors 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. 5.5, 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. 5.5(a) and 5.5(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., $H \times \text{ceil}(S/N) = 4$), 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 [160] and [215], as discussed next.

Mapping for the bitcount circuit from [160] and [215] (Fig. 5.5(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 5.3.2), 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 $psums$, 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. 5.5(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

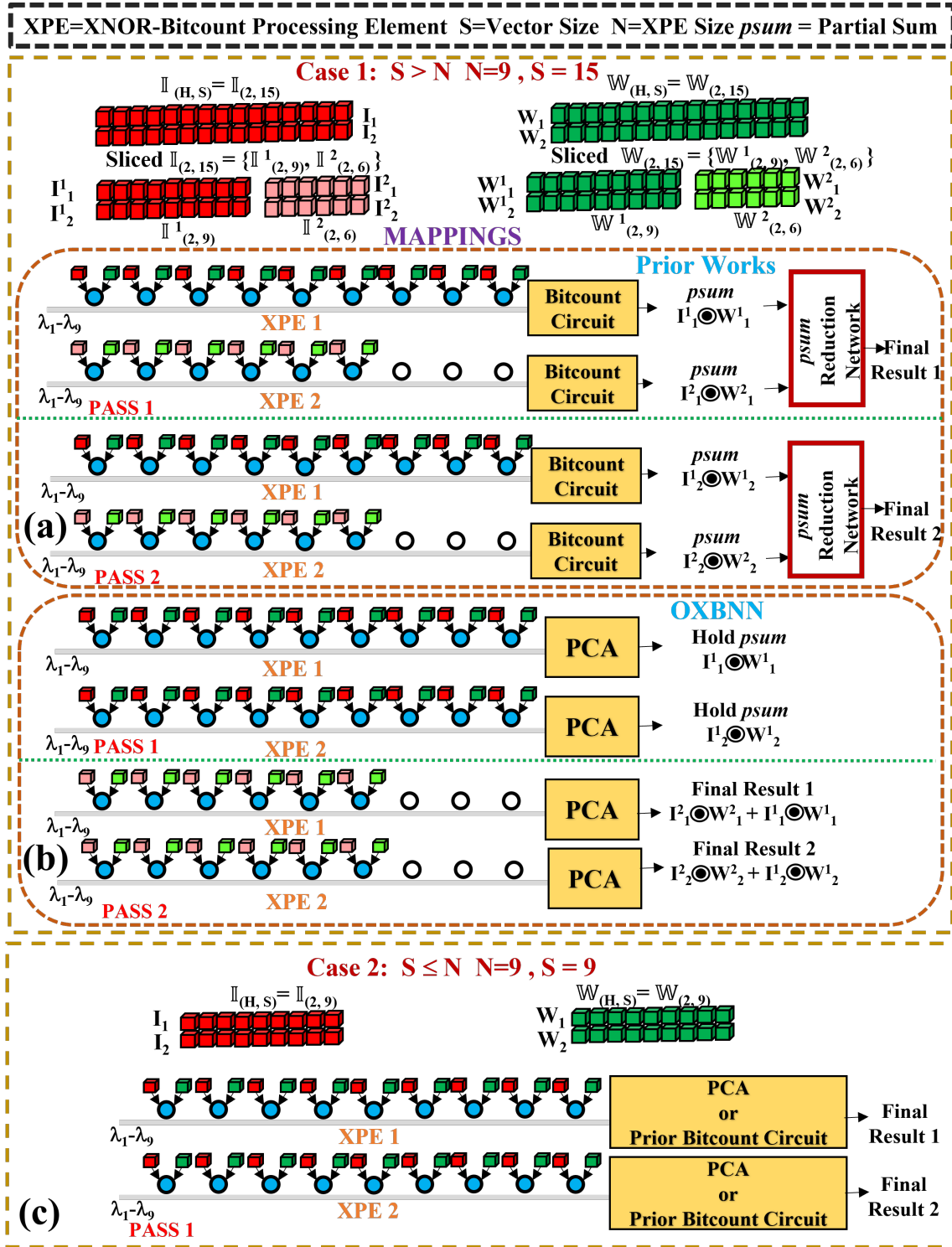


Figure 5.5: Example mappings and related operation of our XPC for various cases of the S and N values. A comparison of our PCA with the bitcount circuit from prior works is also illustrated.

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 5.3.2), 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 5.4.3.

Case 2, $S=9$, $S \leq N$, Fig. 5.5 (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 $I_2 \odot W_2$, respectively. In this case, the mapping is identical for our PCA and the bitcount circuits from prior work.

5.4.3 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 $psums$. From Section 5.4 and Table 5.2, our PCA can achieve $\gamma=8503$ and $\alpha=447$ at $DR=50$ GS/s, which means that our PCA, before it saturates, can accumulate a total of $\gamma=8503$ '1's across a total of $\alpha=447$ 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 $\gamma=8503$. 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, EfficientNet_B7, NASNetMobile, MobileNet_V2, and ShuffleNet) [37], our PCA eliminates the need to employ dedicated $psum$ reduction networks in our OXBNN accelerator.

5.5 Evaluation

5.5.1 System-Level Implementation of OXBNN.

Fig. 5.6 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.

5.5.2 Simulation Setup

For evaluation, we model our OXBNN accelerator from Fig. 5.6 using our custom-developed, transaction-level, event-driven python-based simulator (<https://github.com>).

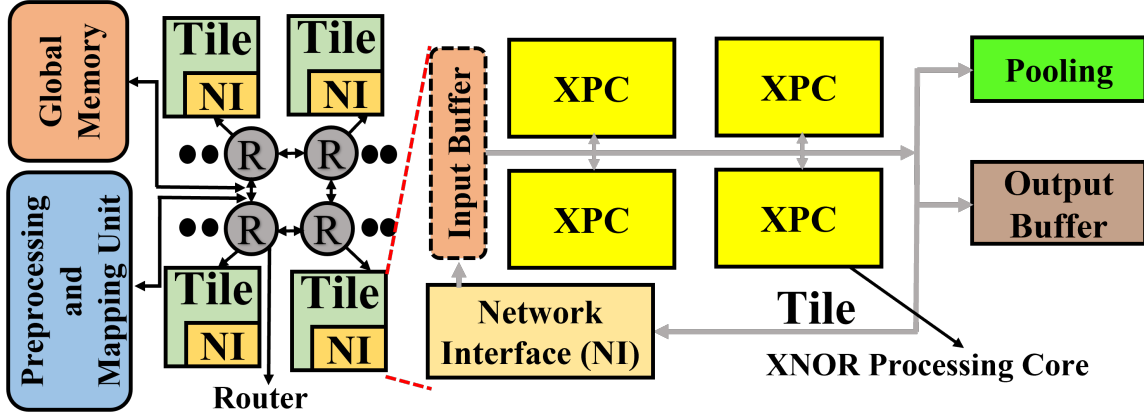


Figure 5.6: System-level overview of our OXBNN accelerator.

Table 5.3: Accelerator Peripherals and XPE Parameters [152]

	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 μ W/FSR	20ns	-
TO Tuning	275 mW/FSR	4 μ s	-

com/uky-UCAT/B_ONN_SIM). We simulated the inference of four BNNs (batch size=1): VGG-small[205], ResNet18[65], MobileNet_V2 [138], and ShuffleNet_V2 [208]. We binarized all the weights and inputs using the LQ-Nets technique[205]. We evaluate frames-per-second (FPS) and FPS/W (energy efficiency).

We compared our OXBNN with ROBIN [160] and LIGHTBULB [215]. 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 5.2), (2) OXBNN_50 with DR=50GS/s (matching with LIGHTBULB) and $N=19$ (Table 5.2). We consider two variants of ROBIN: ROBIN Energy-Optimized (ROBIN_EO) and ROBIN Performance-Optimized (ROBIN_PO)[160]. 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 5.3 gives the parameters used for our evaluation.

5.5.3 Evaluation Results

Fig. 5.7(a) compares FPS values (log scale). OXBINN_50 achieves $62\times$, $8\times$, and $7\times$ 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 $54\times$, $7\times$, and $16\times$, 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).

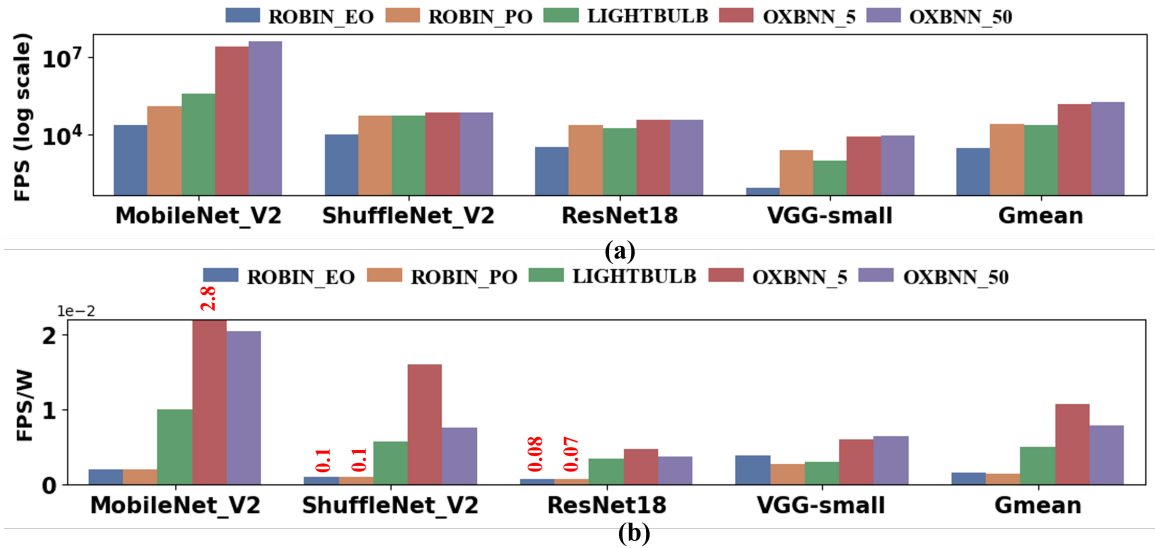


Figure 5.7: (a) FPS (log scale) (b) FPS/W for OXBNN versus ROBIN and LIGHTBULB accelerators.

Fig. 5.7(b) gives the energy efficiency (FPS/W) achieved by each accelerator across various BNNs. Our OXBINN_5 gains $6.8\times$, $7.6\times$, and $2.14\times$ 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 $4.9\times$, $5.5\times$, and $1.5\times$, 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 5.4.3) also eliminates related high energy consumption. Thus, these benefits collectively render better FPS/W for OXBNN_5 and OXBNN_50.

5.6 Summary

In this Chapter, 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 $62\times$ and $7.6\times$ 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 Chapter. We would also like to acknowledge the National Science Foundation (NSF) as this research was supported by NSF under grant CNS-2139167.

Chapter 6 A Stochastic Computing Based Optical Accelerator for Ultra-Fast, Energy-Efficient Inference of Integer-Quantized CNNs

6.1 Introduction

Deep Neural Networks (DNNs) have revolutionized the implementation of various artificial intelligence tasks, such as image recognition, language translation, autonomous driving [96, 47], due to their high inference accuracy. Convolutional Neural Networks (CNNs) are specific types of DNNs [105]. CNNs are computationally intensive, and hence, require a long inference time. In CNNs, around 80% of the total processing time is taken by convolution operations that can be decomposed into vector dot product (VDP) operations [198]. The ever-increasing complexity of CNNs has pushed for highly customized CNN hardware accelerators [20]. Often, for efficient and swift hardware-based acceleration, CNNs are typically quantized to have integer input/weight parameters [91]. Among CNN hardware accelerators, silicon-photonics accelerators have shown great promise to provide unparalleled parallelism, ultra-low latency, and high energy efficiency [109, 58, 22, 34, 201, 157]. Typically, a silicon-photonics CNN accelerator consists of multiple Vector Dot Product Cores (VDPCs) that perform multiple VDP operations in parallel. Several VDPC-based optical CNN accelerators have been proposed in prior works based on various silicon-photonics devices, such as Mach Zehnder Interferometer (MZI) (e.g., [16], [206], [44]) and Microring Resonator (MRR) (e.g., [22], [157], [164], [166]).

Among these optical VDPC-based CNN accelerators from prior work, the MRR-enabled VDPC-based accelerators (e.g., [109, 22, 58, 145, 157, 166]) have shown disruptive performance and energy efficiencies, due to the MRRs' compact footprint, low dynamic power consumption, and compatibility with cascaded dense-wavelength-division-multiplexing (DWDM). Among these MRR-enabled accelerators, some accelerators utilize digital VDPCs (e.g., [145]), whereas some others employ analog VDPCs (e.g., [157, 22, 166]). In general, a VDPC (analog or digital) transforms convolution operations into vector dot product (VDP) operations by decomposing the input tensors into vectors (1D tensors). In an analog VDPC, such VDP operations are also analog in nature, and they are performed on the individual VDP elements (VDPEs), which are the main MRR-enabled hardware components in the VDPCs. Multiple VDPEs in an analog VDPC can perform multiple analog VDP operations in parallel. The results of these analog VDP operations are converted into the digital format using analog-to-digital converters (ADCs). These results can be summed together (if and when needed) using a partial-sum (*psum*) reduction network, which can be employed outside of the VDPCs as part of the post-processing components of the CNN accelerator. The functioning of the analog VDPCs and their constituent VDPEs in the ultra-high-speed, analog-optical domain results in disruptive throughput for performing analog VDP operations.

We observe that two factors govern the performance of such analog optical VDPCs: (1) the achievable bit-precision (B) and (2) the achievable scalability of the VDPCs,

i.e., the achievable count of the individual VDPEs per VDPC (M) and the individual VDPE size (the number of multiplications that can be generated and summed up per VDPE) (N). In an analog VDPC, the achievable B affects the inference accuracy of the processed CNNs, whereas the achievable VDPC scalability (i.e., N and M) directly defines the throughput of the VDPC for processing CNNs. Prior works [9] and [70] studied various factors such as optical power budget in waveguides, inter-channel spacing of wavelengths, crosstalk at cascaded MRRs, resolution of ADCs, and photodetector responsivity, to determine the bounds of the achievable B and scalability in analog optical VDPCs. Furthermore, prior work [152] characterized the very strong trade-off between the maximum achievable VDPC size N and B in analog optical VDPCs. From [152], the analog optical VDPCs from prior works cannot support N greater than 44 for $B \geq 4$ -bit [152]. Achieving such low N can seriously hurt the performance for processing modern CNNs. This is because modern CNNs employ tensors with as high as 4608 points (parameters) per tensor [65]. Processing such large tensors on a VDPC with $N \leq 44$ results in a large number of *psums*, resulting in a very high latency overhead in the *psum* reduction network.

To avoid this undesired outcome, we advocate for such an architecture of MRRs-based CNN accelerator that achieves significantly larger VDPC size N along with weakened interdependence between N and B . To that end, for the first time, we leveraged the inherent precision flexibility of stochastic computing to come up with a novel, MRRs-enabled **Stochastic Computing based Optical Neural Network Accelerator** (SCONNA). SCONNA employs our invented MRR-based Optical Stochastic Multipliers (OSMs) to realize manifold improvements in the throughput and energy efficiency of processing integer-quantized CNNs.

Our key contributions in this Chapter are summarized below:

- To enable stochastic computing in the optical domain, we present (i) a novel design of optical stochastic multiplier (OSM), and (ii) a novel photo-charge accumulator (PCA) circuit (Section 6.4);
- We present detailed modeling and characterization of our invented OSM and PCA using foundry-validated, commercial-grade, photonic-electronic design automation tools (Section 6.4);
- We employ our designed OSMs and PCAs to forge a highly scalable CNN accelerator named SCONNA, which employs OSM and PCA-based scalable VDPCs (Section 6.4);
- We perform a comprehensive scalability analysis for our SCONNA VDPCs, to determine their achievable maximum size N , operating speed, and error susceptibility (Section 6.5);
- We implement and evaluate SCONNA at the system-level using our in-house simulator (https://github.com/uky-UCAT/SC_ONN_SIM.git), and compare its performance and inference accuracy for processing 8-bit integer-quantized CNNs with two widely-known MRR-based analog CNN accelerators from prior works (Section 6.6).

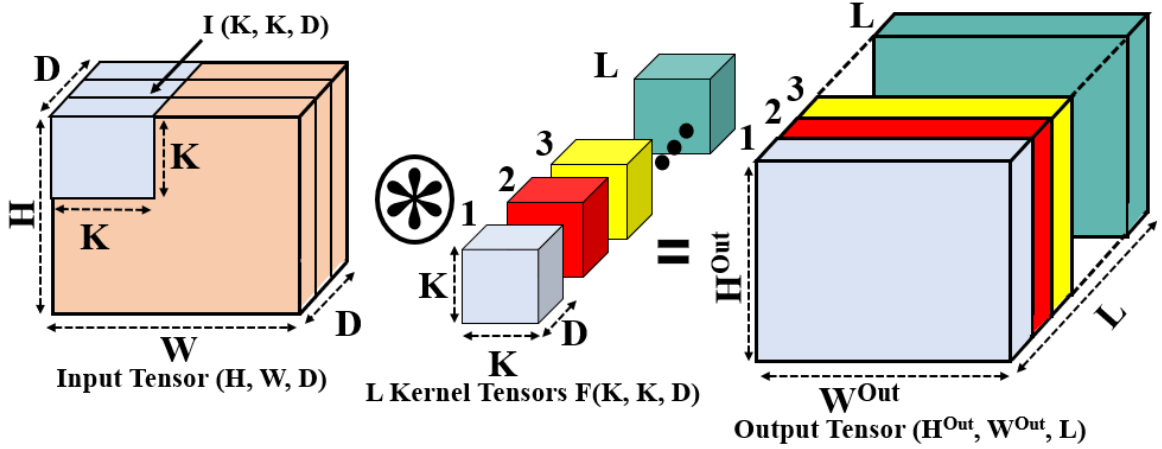


Figure 6.1: Illustration of a convolution operation.

6.2 Preliminaries

6.2.1 Convolutional Neural Networks (CNNs)

CNNs are specific types of DNNs that have shown remarkable accuracy for image classification. In general, a CNN consists of multiple convolutional layers, pooling layers, and fully connected layers. As shown in Fig. 6.1, a typical convolutional layer consists of one input tensor $\mathcal{I}(H, W, D)$ and L kernel tensors $\mathcal{F}(K, K, D)$. All of the L kernel tensors convolve over the input tensor using stride (ψ) to produce the output tensor $\mathcal{O}(H^{Out}, W^{Out}, L)$.

The computation required to produce each point $O(i, j, l)$ in the output tensor $\mathcal{O}(H^{out}, W^{out}, L)$ can be given as Eq. 6.1.

$$O(i, j, l) = \sum_{d=1}^D \sum_{q=1}^K \sum_{r=1}^K F(r, q, d) I(i \times \psi + r, j \times \psi + q, d) \quad (6.1)$$

Here, $d=[1, D]$, $q=[1, K]$, $r=[1, K]$, $i=[1, H^{Out}]$, $l=[1, L]$, and $j=[1, W^{Out}]$ are various indices and their value ranges for the kernel and output tensors. $O(i, j, l)$ in Eq. 6.1 is the sum of a total of $K \times K \times D$ products (products of the individual points of tensors \mathcal{F} and $\mathcal{I}(K, K, D)$; $\mathcal{I}(K, K, D)$ is the gray-highlighted part of $\mathcal{I}(H, W, D)$ in Fig. 6.1). Thus, producing $O(i, j, l)$ requires $K \times K \times D$ point-wise multiplications (to produce $K \times K \times D$ point-wise products) and one sum-of-products operation. The combination of these point-wise multiplications and the corresponding sum-of-products operation is mathematically equivalent to a Vector Dot Product (VDP) operation. A VDP operation typically occurs between two vectors. This implies that I and F in Eq. 6.1 are vectors, which are basically flattened (in 1D) versions of tensors $\mathcal{I}(K, K, D)$ and $\mathcal{F}(K, K, D)$ respectively. Note that vectors I and K have a total of $S = K \times K \times D$ points each. Henceforth, We refer to I and K as input vector and kernel vector, respectively.

6.2.2 Processing Convolutions on VDPCs

Producing the output tensor $\mathcal{O}(H^{Out}, W^{Out}, L)$ (Fig. 6.1) requires that the VDP operation shown in Eq. 6.1 is implemented multiple times, i.e., a total of $H^{Out} \times W^{Out} \times L$ times. In Eq. 6.1, the output $O(i,j,l)$ is the result of the VDP operation between the corresponding input vector and kernel vector, each of size $S = K \times K \times D$ (Section II.A). Typically, for a CNN, the values K and D vary dramatically across different kernel tensors of the CNN. Therefore, $S = K \times K \times D$ also varies dramatically. The value S for CNNs can be as large as 4608 (e.g., ResNet50 [152]). Because of such large S , to accelerate VDP operations on a VDPC, it is intuitive to have the size N of the constituent VDPEs of the VDPC (defined as the number of point-wise multiplications a VDPE can concurrently perform) to be as large as S . However, it is hardly possible to have N to be equal to S in optical MRR-based analog VDPCs. Therefore, input vector and kernel vector are generally divided into multiple decomposed input vectors (*DIVs*) (this and other abbreviations are defined in Table 6.3) and decomposed kernel vectors (*DKVs*) first, and then these *DIVs* and *DKVs* are processed on the VDPEs (Section III.A). Having to decompose the input vector and kernel vector into multiple *DIVs* and *DKVs* raises several challenges as discussed in Section III.A.

6.2.3 Optical Analog VDPC-Based CNN Accelerators

Most of the optical MRR-enabled analog, incoherent CNN accelerators from prior work employ multiple optical analog VDPCs that work in parallel. A brief review of prior works on optical accelerators is provided in Section 6.7. Typically, an analog VDPC implements the decomposed VDP operations of a convolution operation using *DKVs* and *DIVs* (Section II.A). In general, a VDPC consists of five blocks (Fig. 6.2(a)): (i) a laser block that consists of N laser diodes (LDs) to generate N optical wavelength channels; (ii) an aggregation block that aggregates the generated optical wavelength channels into a single photonic waveguide through dense wavelength division multiplexing (DWDM) (using an $N \times 1$ multiplexer) and then splits the optical power of these N wavelength channels equally into M separate waveguides (using a $1 \times M$ splitter); (iii) a modulation block, also referred to as *DIV* block, that employs M arrays of MRRs (one array per waveguide, with each array having N MRRs; each array referred to as *DIV* element) to imprint M *DIVs* of N points each onto the $N \times M$ wavelength channels by modulating the analog power amplitudes of the wavelength channels; (iv) another modulation block, referred to as *DKV* block, that employs another M arrays of MRRs (one array per waveguide, with each array having N MRRs; each array referred to as *DKV* element) to further modulate the $N \times M$ wavelength channels with *DKVs*, so that the analog power amplitudes of the individual wavelength channels then represent the point-wise products of the utilized *DKVs* and *DIVs*; and (v) a summation block (SB) that employs a total of M summation elements (SEs), with each SE having two balanced photodiodes (PDs) upon which the point-wise-product-modulated N wavelength channels are incident to produce the output current that is proportional to the result of the VDP operation between the corresponding *DKV* and *DIV*. The laser block and SB are typically positioned at the

two ends of the VDPC, with the aggregation, modulation (*DIV*), and modulation (*DKV*) blocks placed in between them.

Based on the order in which these intermediate blocks (aggregation, modulation (*DIV*), modulation (*DKV*) blocks) are positioned between the laser block and SB, we classify the MRR-based VDPC organizations from prior work as MAM (Modulation, Aggregation, Modulation) (e.g., [109], [9]) or AMM (Aggregation, Modulation, Modulation) (e.g., [201], [58], [22]). Fig. 6.2 illustrates MAM and AMM VDPC organizations. From Fig. 6.2(a), the AMM VDPC organization positions the aggregation block first after the laser block, and then the *DIV* modulation block followed by the *DKV* modulation block. In contrast, the MAM VDPC in Fig. 6.2(b) positions the *DIV* modulation block first after the laser block, and then positions the aggregation block followed by the *DKV* modulation block. Note that the MAM *DIV* block is structurally different from the AMM *DIV* block. The MAM *DIV* block employs only one MRR per waveguide, and as a result, it can imprint only one *DIV* with N points onto the N wavelength channels. This one *DIV* is shared among all *DKVs* in the MAM VDPC, whereas each *DKV* can have a different *DIV* corresponding to it in the AMM VDPC. Most MAM and AMM VDPCs from prior works have $M=N$.

In both the AMM and MAM VDPC organizations, we refer to the combination of a *DKV* element and the corresponding SE as VDP element (VDPE). However, the size and point-wise product precision of MRR-based VDPEs have certain limitations (discussed in Section 6.3). These limitations demand exploration of new computing options to improve MRR-based VDPCs, and stochastic computing is an attractive option.

6.2.4 Stochastic Computing

Stochastic Computing (SC) is an unconventional form of computing that represents and processes data in the form of probabilistic values called stochastic numbers (SNs) [54, 10, 15]. In SC’s unipolar format, an SN W is a bit-stream of N bits that represents a real-valued variable $v \in [0, 1]$ by encoding v through the ratio N_1/N , where N_1 is the number of 1’s in W . SC offers several advantages over conventional binary computing such as high error tolerance, low power consumption, small circuit area, and low-cost arithmetic operations consisting of standard digital logic components [15]. For example, multiplication can be performed by a stochastic circuit consisting of a single AND gate.

Fig. 6.3 illustrates a multiplication between two unipolar stochastic bit-streams I and W using an AND gate. The probabilities of seeing ‘1’s in the bit-streams I and W are (4/8) and (6/8), respectively. The AND gate performs bit-wise logical AND operation on the bit-streams to produce the output bit-stream A . In A , the probability of seeing ‘1’s is (3/8), which is equal to (4/8)×(6/8), i.e., the multiplication (or product) of the input probabilities. Note that for the AND gate to produce an error-free multiplication output, the marginal probability of one bit-stream (i.e., I or W) should be equal to its conditional probability given the other bit-stream (i.e., I given W or W given I)[196]. Also note that, because of its advantages, SC has been adopted in stochastic deep CNNs [133, 100, 104], GEMM computation [196], and

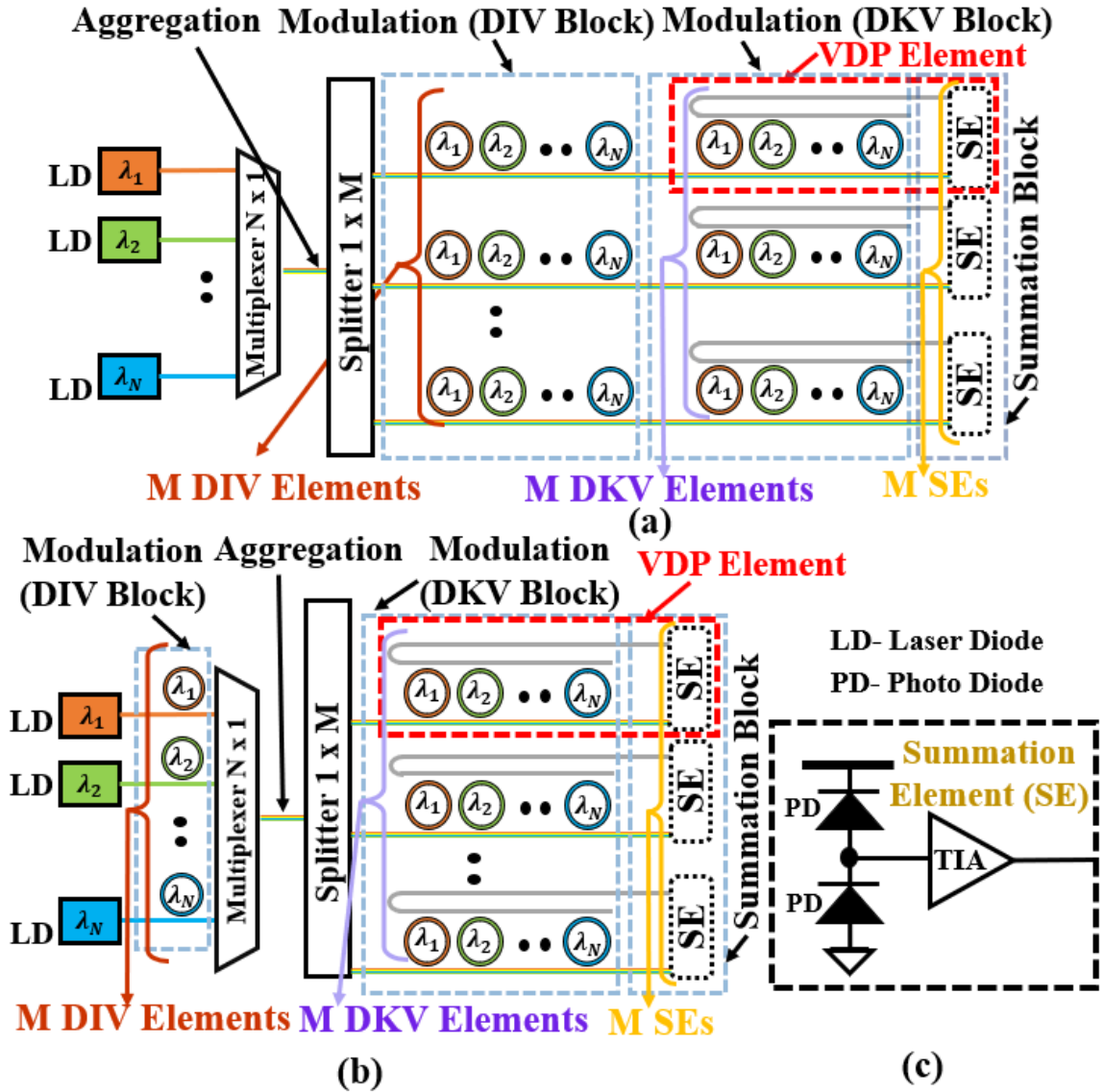


Figure 6.2: Illustration of common analog optical VDP organizations: (a) AMM VDP, (b) MAM VDP. (c) Summation Element.



Figure 6.3: Multiplication between unipolar stochastic numbers I and W .

image processing [101]. We use stochastic computing in this Chapter to relax the inherently strong scalability-precision trade-off in the optical VDPs. This trade-off

is explained in the next section.

6.3 Motivation

6.3.1 Scalability Limitations of MRR-Based Analog VDPCs

Prior works [94], [152], and [9] have analysed the scalability (i.e., achievable value of VDPE size N under the constraints of bit precision and data rate) of AMM and MAM VDPCs. Table 6.1 reproduces the supported values of VDPE size N (considering $M=N$) for AMM and MAM VDPCs at various data rates (DRs) and bit precision from [152]. From Table 1, the maximum $N=44$ is obtained for MAM VDPC across all tested DR and B values. For MAM VDPC for 1 GS/s, maximum N reduces from 44 to 12 as we increase the input/weight precision from 4-bit to 6-bit. The reason for such strong trade-off between N and achievable input/weight precision (referred to as B , henceforth) in MAM and AMM VDPCs is that both B and N strongly depend on the number of distinguishable analog optical power levels [152][94], which is proportional to $N \times 2^B$. Hence, for a fixed number of distinguishable analog optical power levels, which is defined by the analog optical power resolution of the utilized summation elements (SEs) (see SEs in Fig. 6.2) the supported N drastically decreases with an increase in B . As a result, N decreases all the way to 1 when B increases to 8-bit [152].

Due to such strong trade-off between N and B , the MAM and AMM type of analog VDPCs face two consequences. First, they produce high number of partial sums and incur significantly high latency for partial sum reduction. For example, a VDPE with $N=44$ for $B=4$ -bit can only produce a VDP operation between two 44-point vectors. Therefore, producing a VDP operation between an input vector and kernel vector with $S=4608$ (e.g., ResNet50 [65][152]) requires that the input vector is first decomposed into a total of $C=\text{Ceil}(S/N)=105$ *DIV*s of $N=44$ points each. Similarly, the kernel vector also needs to be decomposed into a total of $C=\text{Ceil}(S/N)=105$ *DKV*s of $N=44$ points each. Then, a total of 105 VDPEs can be employed to perform 105 VDP operations between 105 pairs of *DKV*s and *DIV*s, to consequently produce a total of 105 intermediate VDP results (i.e., partial sums (*psums*)). Although these 105 VDP operations can be parallelized over 105 VDPEs, producing the final VDP result of $S=4608$ would require the accumulation of the 105 *psums*. Doing so can incur very high latency and energy consumption, which should be avoided using a more efficient VDPC design.

As the second consequence, the throughput of the MAM and AMM VDPCs decreases at higher bit precision (higher value of B). This is because to avoid a drastic decrease in N as B increases beyond 4-bit, the AMM and MAM type of analog VDPCs typically operate at $B=4$ -bit [152]. However, using at least 8-bit precision ($B=8$ -bit) for the integer-quantized CNN models is recommended to achieve competitive inference accuracy, while also reducing the computational effort, memory requirements, and energy consumption [91]. To meet this requirement, analog VDPCs from [109] (an MAM VDPC) and [22] (an AMM VDPC) employ bit-slicing of input/weight parameters. They slice each 8-bit integer input/weight parameter into two slices of 4-bit

Table 6.1: VDPE size N for input/weight precision= $\{4,6\}$ -bit at data rates (DRs)= $\{1,3,5,10\}$ GS/s, for AMM and MAM VDPCs.

VDPC	Precision	Datarate(DR)			
		1 GS/s	3 GS/s	5 GS/s	10 GS/s
AMM	4-bit	31	20	16	11
	6-bit	6	3	2	1
MAM	4-bit	44	29	22	16
	6-bit	12	7	5	3

Table 6.2: Total number of kernels (T_L) of different DKV sizes (S) for various CNNs. The T_L values were extracted for trained CNN models from Keras Applications [37].

Model	T_L	S	Model	T_L	S
ResNet50	1	$S \leq 44$	GoogleNet	13	$S \leq 44$
	26562	$S > 44$		7554	$S > 44$
VGG16	69	$S \leq 44$	DenseNet	1	$S \leq 44$
	4168	$S > 44$		10242	$S > 44$

each. Then, they employ two VDPCs in parallel; each VDPC processes one 4-bit slice of the input/weight parameters. The corresponding 4-bit VDP results from these two 4-bit VDPCs are then combined using shifters and adders to produce the final 8-bit results. Thus, performing VDP operations using bit slices reduces the total number of VDP results that can be produced by a fixed number of VDPCs, because multiple VDPCs are needed to produce a single set of VDP results. This can severely degrade the throughput of such VDPCs. Such undesired outcome should be avoided by designing a more efficient VDPC.

6.3.2 Need for Stochastic Computing

Table 6.2 reports the counts of kernel tensors according to their sizes S ($S \leq 44$ and $S > 44$) for four modern CNN models. From Table 6.2, more than 98% of the kernel tensors across all four CNNs have $S > 44$, and thus, they require VDPEs with size $N > 44$ to process their corresponding VDP operations. But, from Table 6.1, the maximum achievable N for analog VDPCs at 4-bit precision ($B=4$ -bit) is limited to 44; therefore, processing the VDP operations corresponding to more than 98% of kernel tensors that have $S > 44$ would lead to high *psum* reduction latency (see Section 6.3.1). However, reducing this *psum* reduction latency in analog VDPCs is challenging, as they have a strong trade-off between N and B , and this is because the required number of analog optical power levels (i.e., 2^B) to support a specific B consumes a large part of the available dynamic range of optical power in analog VDPCs. To this end, the remaining dynamic range of optical power within the total allowable optical power budget restricts the supported N in analog VDPCs. This limitation can be addressed by performing VDP operations in the digital domain [145]. There is no need to support any analog optical power levels in the digital domain;

therefore, most of the available dynamic range of optical power in a digital VDPC can be used to support a higher N . But, the MRR-based binary digital VDPCs (e.g., [145, 85]) suffer from very high hardware complexity, and their multiply-accumulate and bit-shifting circuits consume huge area. These drawbacks motivate the need to examine new options for realizing optical digital VDPCs.

One such option is stochastic computing. In stochastic computing, multiplications can be replaced with simple bitwise AND operations [15]. This can be leveraged to perform point-wise multiplications between $DKVs$ and $DIVs$ (Section 6.2.3) with less hardware complexity than binary digital VDPCs. In addition, since stochastic computing is also implemented in the digital domain (non-binary), a stochastic computing based optical VDPC can support a large N due to the large available dynamic range of optical power, just as discussed above for a binary digital VDPC. Moreover, a stochastic computing based optical VDPC can attain different precision levels by merely changing the number of bits in the stochastic bit-streams, without requiring different analog optical power levels. Therefore, to utilize these advantages of stochastic computing, prior works [209] and [49] proposed stochastic computing based photonic acceleration. [209] reports acceleration of Markov Random Field Inference and [49] employs photonic crystals and MZIs to build an edge detection filter. However, none of the prior works have employed stochastic computing based photonic acceleration for neural network inference. To fill this gap, we invent an MRR-based optical stochastic multiplier (OSM) and employ multiple OSMs to forge a novel Stochastic Computing Optical Neural Network Accelerator (SCONNA). The following section discusses our SCONNA architecture.

6.4 Our Proposed SCONNA Architecture

6.4.1 Overview of SCONNA VDPC

Fig. 6.4(a) illustrates the VDPC organization of our SCONNA architecture. Like the VDPCs of analog optical accelerators, a SCONNA VDPC also implements multiple VDP operations in parallel. For that, an array of total N single-wavelength laser diodes (LDs) are used, 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 (λ_1 to λ_N) multiplexed into a single photonic waveguide through wavelength division multiplexing (WDM). These multiplexed wavelengths split into M input waveguide arms (IWAs). Every IWA receives N -wavelength optical power and guides it to a VDPE. There are a total of M IWAs and M VDPEs in the SCONNA VDPC (Fig. 6.4(a)).

Each VDPE consists of three components: (i) a cascade of N Optical Stochastic Multipliers (OSMs); (ii) a bank of filter MRRs; (iii) a Photo-Charge Accumulator (PCA) pair. Each OSM performs stochastic multiplication between an input bit-stream I (corresponding to a point in an N -point DIV) and weight bit-stream W (corresponding to a point in an N -point DKV). Each OSM receives its bit-streams I and W from its corresponding peripherals at a supported bitrate (BR). Bit-stream W provides a weight value along with a sign bit. Bit-stream I provides the RELU-activated output value from the previous CNN layer, without a sign bit as RELU has a

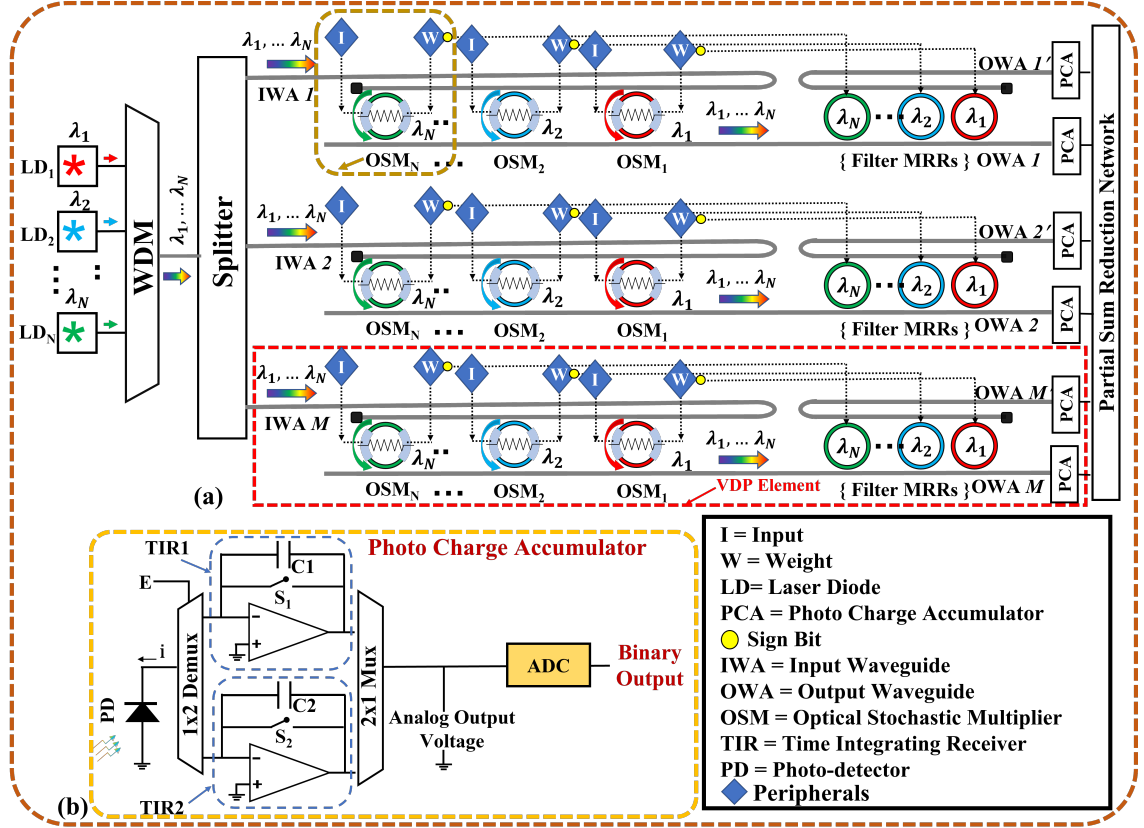


Figure 6.4: Schematics of (a) Our SCONNA VDPC (b) Photo-Charge Accumulator (PCA) Circuit.

non-negative output. The detailed design of *OSMs* and their peripherals is explained in Section 6.4.2. Each *OSM* performs a bit-wise logical AND operation between the *I* and *W* bit-streams to produce a resultant optical bit-stream that represents the stochastic multiplication between the *I* and *W* bit-streams. The N optical bit-streams from the cascade of N *OSMs*, with each bit-stream carrying a stochastic multiplication result, reach the bank of filter MRRs. In this bank, each filter MRR operates on a distinct optical bit-stream λ_i . Each filter MRR receives the sign bit from the peripheral *W* of its corresponding *OSM* (Fig. 6.4(a)). The sign bit operates the filter to steer the incoming optical bit-stream λ_i to the output waveguide arm *OWA* (if the sign bit is '0') or *OWA'* (if the sign bit is '1'). Thus, the *OWA* and *OWA'* of a VDPE guide the optical bit-streams, carrying the stochastic multiplication results, to PCAs. A PCA is a circuit that collects all the optical bit-streams (i.e., stochastic multiplication results) from its corresponding *OWA* (or *OWA'*) and generates the accumulation result in the binary format (details about PCA in Section 6.4.3). In a VDPE, the *OWA*-coupled PCA combines with the corresponding *OWA'*-coupled PCA to generate a signed accumulation result.

6.4.2 Optical Stochastic Multiplier

Our Optical Stochastic Multiplier (OSM) consists of peripherals and an Optical 'AND' Gate (OAG) (Fig. 6.5). The peripherals convert a binary input value I_b and binary weight value W_b into unipolar stochastic bit-streams I and W , and OAG performs multiplication-equivalent bitwise AND operation between the stochastic bit-streams I and W .

From Fig. 6.5, the peripherals of our OSM use a lookup table and serializers to generate a combination of unipolar stochastic bit-streams I and W . From [196], two unipolar stochastic bit-streams, for their eventual error-free multiplication using an AND gate, should be generated in combination with each other, so that they are uncorrelated, i.e., the marginal probability of one bit-stream (i.e., I or W) is equal to its conditional probability given the other bit-stream (i.e., I given W or W given I). For our OSM, we propose to generate all possible combinations of uncorrelated bit-streams I and W a priori (offline) using the unipolar circuit from [196], and then store these bit-streams in the bit-vector (bit-parallel) format in the lookup table (Fig. 6.5). As a result, each entry in the lookup table stores a combination of uncorrelated bit-vectors I_v and W_v . To index into this lookup table, our OSM creates a unique identifier for each combination of binary values I_b and W_b (that are accessed from a buffer (a scratchpad memory); Fig. 6.5) by performing an XOR-based hash function $I_b \oplus W_b$. Thus, our OSM uses a $I_b \oplus W_b$ value to fetch the desired combination of I_v and W_v from the lookup table. Then, it pushes these I_v and W_v through dedicated high-speed serializers, to generate bit-streams I and W . Lookup table size: If precision $B=8$ -bit for binary I_b and W_b , there are 2^B entries in the lookup table, with each entry storing two 2^B -bits long bit-vectors.

The stochastic bit-streams I and W , generated by the peripherals of our OSM, are then fed to the OAG via high-speed drivers for their stochastic multiplication (Fig. 6.5). The design of our OAG is illustrated in Fig. 6.6(a). Our OAG is an add-drop microring resonator (MRR), which has two operand terminals (realized as embedded PN-junctions) that can take two stochastic bit-streams I and W (Fig. 6.6(a)) as inputs at a predefined bitrate (BR). Fig. 6.6(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. 6.6(a)), to consequently tune its operand-independent resonance from its fabrication-defined initial position γ to its programmed position η , relative to the input optical wavelength position λ_{in} (Fig. 6.6(b)). 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 blue passbands in Fig. 6.6(b)). Based on the MRR resonance passband's programmed position η relative to λ_{in} , the drop-port transmission ($T(\lambda_{in})$) of the MRR provides bit-wise logical AND operation between the inputs I and W .

To validate our OAG, we performed transient analysis with two pseudo-random numbers as shown in Fig. 6.6(c). For that, we modelled and simulated our OAG using the foundry-validated tools from the Ansys/Lumerical's DEVICE, CHARGE, and INTERCONNECT suites [6]. Fig. 6.6(c) shows two input bit-streams (I, W) applied to the two PN junctions of our OAG at BR=10 Gbps. By looking at the

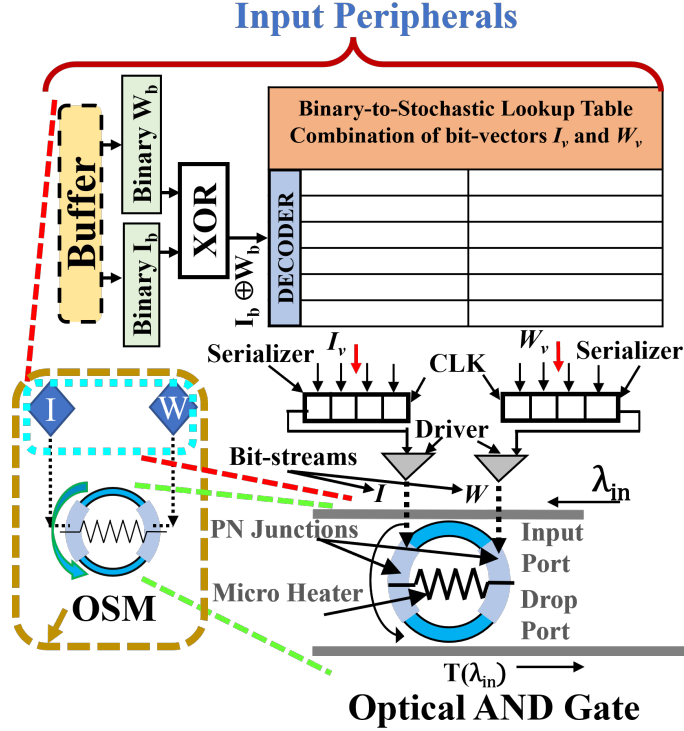


Figure 6.5: Schematic of our Optical Stochastic Multiplier (OSM).

output optical bit-stream $T(\lambda_{in})$ in Fig. 6.6 (c), we can say $T(\lambda_{in}) = I \text{ AND } W$, which validates the functionality of our OAG as a logical AND gate. Thus, since the input bit-streams I and W are in the unipolar stochastic format, the output optical bit-stream at the drop port of the OAG provides the unipolar stochastic multiplication between I and W .

6.4.3 Photo Charge Accumulator (PCA)

From Section 6.4.1, the stochastic multiplication bit-streams generated by OSMs are guided to a PCA, where they are accumulated to generate a binary output value equivalent to the VDP result. Our PCA is inspired from the time integrating receiver (TIR) design from [150] and the photodetector-based optical-pulse accumulator design from [29]. A PCA circuit, shown in Fig 6.4(c), has two stages: (i) a stochastic-to-analog conversion stage; (ii) an analog-to-binary conversion stage. The stochastic-to-analog stage employs a photodetector and two TIR circuits (one TIR circuit remains redundant, enabled by the demux and mux; Fig 6.4(b)). The photodetector generates a current pulse for each optical logic ‘1’ incident upon it. This current pulse accumulates a certain amount of charge on the capacitor of the active TIR circuit (e.g., the circuit with C1 capacitor); as a result, the capacitor accrues an analog voltage level. Hence, when one or more output optical bit-streams are incident upon the photodetector, the total accumulated charge (and thus, the accrued analog

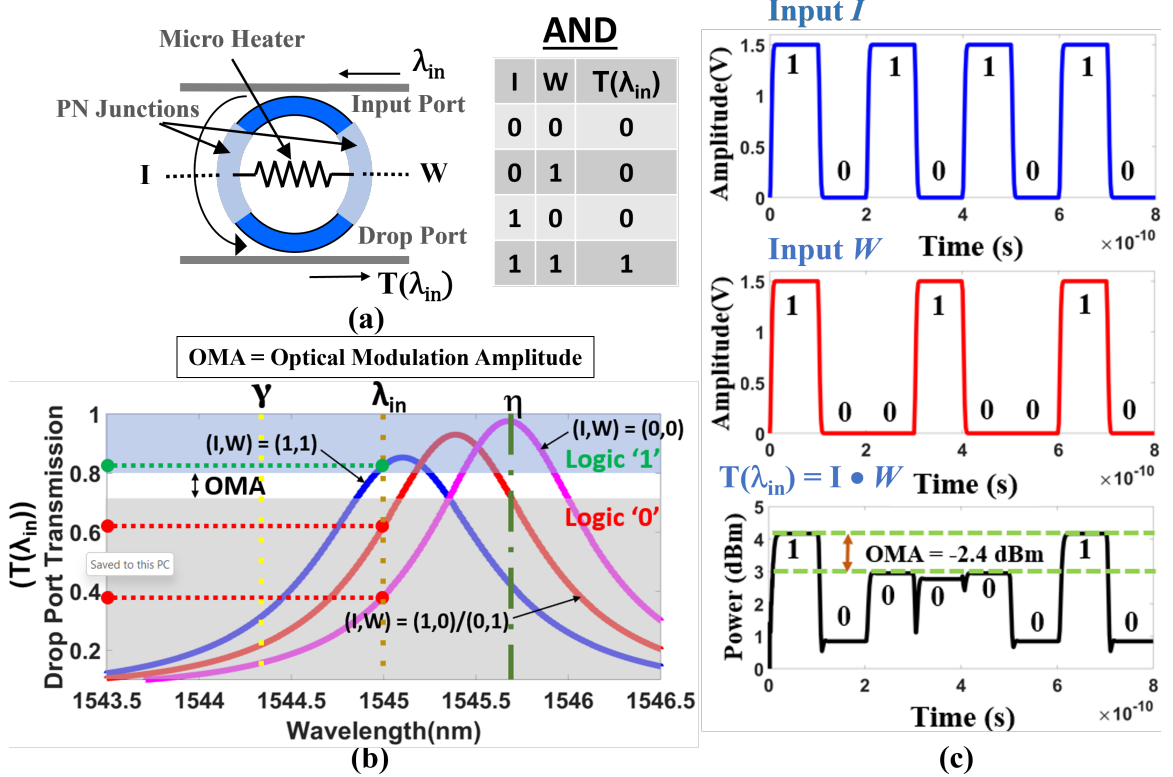


Figure 6.6: (a) Schematic of our Optical AND Gate (OAG), (b) operation of OAG, (c) results of OAG’s transient analysis.

voltage level) on the active capacitor (e.g., C1) is proportional to the total number of ‘1’s in the incident bit-streams. The number of 1’s that can be accumulated in such manner might be limited, as the charge across the capacitor of TIR circuit (Fig. 6.4(b)) might saturate (this is further analysed in section 6.5.3). Once the TIR output saturates, 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, capacitor C2 of the redundant TIR circuit mitigates the discharge latency by allowing a continuation of a concurrent accumulation phase. The output analog voltage computed by the stochastic-to-analog conversion stage represents the unipolar unscaled addition [196] of the stochastic bit-streams. To convert this analog voltage into a binary value, the analog-to-binary stage of the PCA circuit employs an analog-to-digital converter (ADC). This binary value is the VDP result.

6.5 Scalability Analysis of SCONNA Architecture

To understand the scalability of our SCONNA architecture, in this section, we analyze the achievable operating speed of the OSMs, achievable size N of the SCONNA VDPC, and the accumulation capacity of the PCA circuits.

6.5.1 Operating Speed and Latency Overhead of OSM

The peripherals of an OSM can incur some latency for accessing the scratchpad buffer and eDRAM-based lookup table. We consider 2ns latency each for the scratchpad buffer [21] and eDRAM-based lookup table [108]. Beyond this latency, the speed of an OSM depends on the achievable operating speed (bit-rate (BR)) of the constituent OAG. Analysis of OAG's BR: For the output optical bit-stream $T(\lambda_{in})$ in Fig. 6.6(c), the optical modulation amplitude (OMA) is the output power difference between the highest logic '0' power level and the lowest logic '1' power level. OMA should be at least equal to or greater than the sensitivity of the photodetector in the PCA circuit, to ensure that the photodetector in the PCA circuit can produce a distinguishably higher-amplitude current pulse for an optical logic '1' bit compared to an optical logic '0' bit. Keeping the OMA to be greater than or equal to the given photodetector sensitivity ($P_{PD-opt}=-28\text{dBm}$; Section 6.5.2) depends on the OAG's BR and FWHM (full passband width at half maximum). Therefore, to analyze this dependency, we simulated BR and FWHM duplets for which $\text{OMA} = -28 \text{ dBm}$, as shown in Fig. 6.7(a). As evident, supported BR increases as FWHM increases. However, at (FWHM $\approx 0.8\text{nm}$), BR saturates at 40 Gbps. Therefore, we aim to operate our OAG at BR $\leq 40\text{Gbps}$ for FWHM $\leq 0.8\text{nm}$.

6.5.2 Achievable Size of SCONNA VDPC

We consider optimistic free-spectral range (FSR) of 50 nm [9] for the constituent MRR-based OAGs of our SCONNA VDPC. In addition, we consider the inter-wavelength gap of 0.25 nm. This allows the N for our SCONNA VDPC to be 200 ($=\text{FSR}/0.25\text{nm}$), theoretically. However, even if we consider FSR=50nm to be practically achievable for our OAGs, achieving $N=200$ for our SCONNA VDPC might not be possible in practice. This is because when we aim to operate our OAGs at a high BR of $\leq 40 \text{ Gbps}$, for FWHM $\leq 0.8 \text{ nm}$, the total power penalty for our SCONNA VDPC might increase significantly owing to the increased impacts of optical crosstalk effects at OSMs, signal truncation at MRR filters, and BR-dependent increase in the photodetector sensitivity [17, 18, 19]. This increase in power penalty can reduce N to be less than 200. Therefore, to determine the achievable N for our SCONNA VDPC at $B=8\text{-bit}$ precision, we adopt the scalability analysis equations (Eq. 6.2, Eq. 6.3, and Eq. 6.4) from [152, 9]. Table III reports the definitions of the parameters and their values used in these equations. Since our SCONNA VDPC processes stochastic bit-streams, which are digital in format, it requires the bit resolution of $B_{Res} = 1\text{-bit}$ in the equations. Moreover, we conservatively choose to operate OSMs/OAGs at BR=30Gbps. We consider $M=N$. We first solve Eq. 6.2 and Eq. 6.3 for data rate (DR)=BR $\cdot 2^B$, to find P_{PD-opt} to be -28 dBm. Then, we solve Eq. 6.4 for N , to find $N=M=176$, which is a very large N compared to analog VDPCs that have $N\leq 44$. Such large N significantly improves the overall throughput and energy

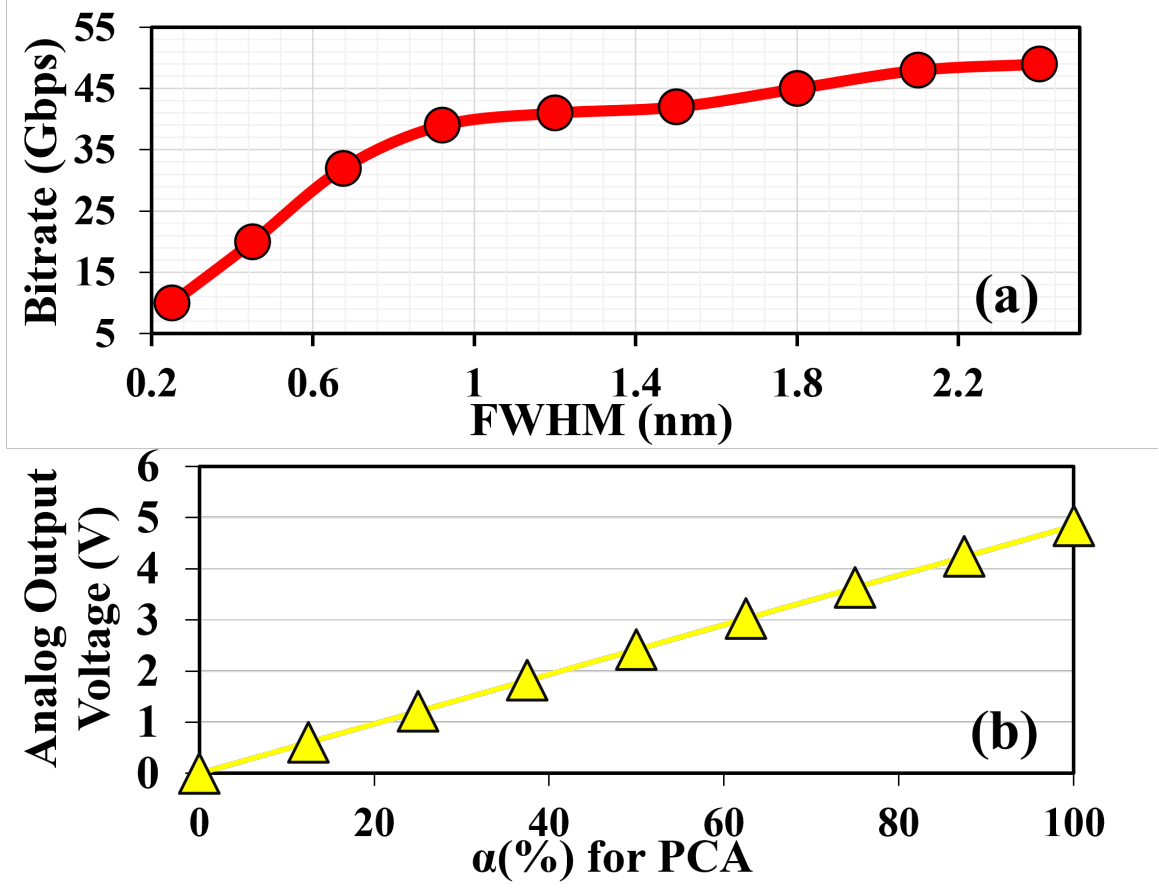


Figure 6.7: (a) Bitrate versus FWHM for our OSM/OAG, (b) Our PCA's analog output voltage versus α

efficiency (Section 6.6).

$$B_{Res} = \frac{1}{6.02} \left[20 \log_{10} \left(\frac{R \times P_{PD-opt}}{\beta \sqrt{\frac{DR}{\sqrt{2}}}}} - 1.76 \right) \right] \quad (6.2)$$

$$\beta = \sqrt{2q(RP_{PD-opt} + I_d) + \frac{4kT}{R_L} + R^2 P_{PD-opt}^2 RIN} \quad (6.3)$$

$$P_{Laser} = \frac{10^{\frac{\eta_{WG}(dB)[N(d_{OSM})]}{10}} M}{\eta_{SMF} \eta_{EC} I L_{i/p-OSM}} \times \frac{P_{PD-opt}}{\eta_{WPE} I L_{MRR}} \times \frac{1}{(OBL_{OSM})^{N-1} (EL_{splitter})^{\log_2 M}} \times \frac{1}{(OBL_{MRR})^{N-1} (IL_{penalty})} \quad (6.4)$$

Table 6.3: List of abbreviations and their full forms used in this Chapter. Definition and values of various parameters (obtained from [9]) used in Eq. 4.3, Eq. 6.3, and Eq. 6.4 for the scalability analysis of our SCONNA VDPCs.

Abbreviations	Full form	Parameter	Definition	Value
VDPC	Vector Dot Processing Core	P_{Laser}	Laser Power Intensity	10 dBm
PCA	Photo Charge Accumulator	R_{PD}	PD Responsivity	1.2 A/W
OAG	Optical AND Gate	R_L	Load Resistance	50 Ω
SE	Summation Element	I_d	Dark Current	35 nA
SC	Stochastic Computing	T	Absolute Temperature	300 K
DKV	Decomposed Kernel Vector	BR	Bitrate	30 Gbps
DIV	Decomposed Input Vector	RIN	Relative Intensity Noise	-140 dB/Hz
VDP	Vector Dot Product	η_{WPE}	Wall Plug Efficiency	0.1
S	Size of DKV	IL_{SMF} (dB)	Single Mode Fiber Insertion Loss	0
$psum$	Partial Sum	IL_{EC} (dB)	Fiber to Chip Coupling Insertion Loss	1.6
OSM	Optical Stochastic Multiplier	IL_{WG} (dB/mm)	Silicon Waveguide Insertion Loss	0.3
DR	Data rate	$EL_{Splitter}$ (dB)	Splitter Insertion Loss	0.01
VDPE	Vector Dot Product Element	IL_{OSM} (dB)	Optical Stochastic Multiplier (OSM) Insertion Loss	4
N	Size of VDPE	OBL_{OSM} (dB)	Out of Band Loss Optical Stochastic Multiplier	0.01
M	Number of VDPEs per VDPC Unit	IL_{MRR} (dB)	Microring Resonator(MRR) Insertion Loss	0.01
OMA	Optical Modulation Amplitude	$IL_{penalty}$ (dB)	Network Penalty	7.3
B	Binary Bit Precision	d_{OSM}	Gap between two adjacent OSMs	20 μ m
B_{Res}	Bit Resolution	PPD_{-opt}	Output Photodetector Sensitivity	-

6.5.3 Accumulation Capacity and Error Susceptibility of PCA

From Section 6.5.2, our SCONNA VDPC has $N=176$. For precision $B=8$, each optical bit-stream in a SCONNA VDPC has $2^B=256$ bits. Therefore, each PCA in a SCONNA VDPC needs to be able to accumulate a total $N \times 2^B=176 \times 256$ optical '1' bits, at the least. We modeled the photodetector of our PCA circuit using the INTERCONNECT tool from Ansys/Lumerical [6] for $R_{PD}=1.2$ A/W and $P_{PD-opt}=-28$ dBm, and extracted the current pulse values corresponding optical '1's and '0's that are consumed by the photodetector. We then imported these values in our MultiSim [2] based model of the TIR circuit of the PCA, to find out that our PCA should have $R=50\Omega$, $C=250$ pF, and voltage amplifier gain=80. For these parameters, we simulated to the analog output voltage at the PCA using MultiSim [2] for different values of $\alpha=(\text{actual \# of '1's in incident bit-streams}/176 \times 256) \times 100\%$. The results are shown in Fig. 6.7(b). As evident, the analog output voltage increases linearly with α without saturating at $\alpha=100\%$. This outcome shows that our PCA can efficiently support the accumulation of $N=176$ bit-streams. Note that the analog output voltage from the amplifier of the PCA circuit does not incur any errors in computation. But, the ADC introduces errors in the generated binary result (we evaluate mean absolute percentage error to be 1.3% for the ADC), and we later evaluate the impact of these errors on the CNN inference accuracy (Section 6.6).

6.6 System-Level Implementation and Evaluation

6.6.1 System-Level Implementation of SCONNA

Fig. 6.8 illustrates the system-level implementation of our SCONNA accelerator. It consists of global memory for storing CNN parameters, and a preprocessing and mapping unit for decomposing the tensors into DIVs/DKVs and mapping them onto VDPEs. It has a mesh of tiles connected to routers, and this mesh network facilitates

Table 6.4: Peripherals Parameters for Accelerators [6].

	Power (mW)	Area (mm ²)	Latency
Reduction Network	0.05	3.00E-05	3.125ns
Activation Unit	0.52	6.00E-04	0.78ns
IO Interface	140.18	2.44E-02	0.78ns
Pooling Unit	0.4	2.40E-04	3.125ns
eDRAM	41.1	1.66E-01	1.56ns
Bus	7	9.00E-03	5 cycles
Router	42	0.151	2 cycles
AMM/MAM			
DAC [82]	30	0.034	0.78ns
ADC [60]	29	0.103	0.78ns
SCONNA			
ADC [122]	2.55	0.002	0.78ns
Serializer per OSM [154]	5	5.9	0.03ns
LUT per OSM [203]	0.06	0.09	2ns
PCA [2]	0.02	0.28	-

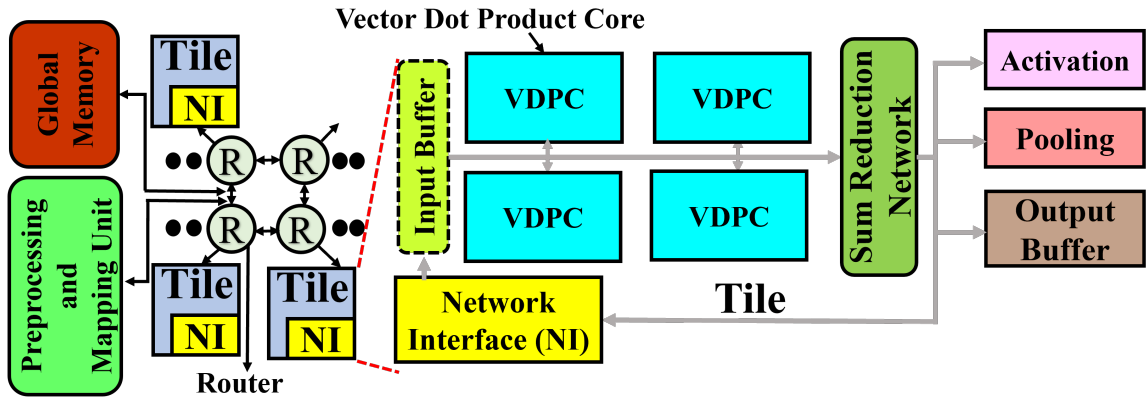


Figure 6.8: System-level overview of our SCONNA CNN accelerator.

parameter communication among tiles. Each tile consists of 4 SCONNA VDPCs interconnected (via H-tree network) with output buffer, activation, and pooling units. In addition, each tile also contains a *psum* reduction network.

6.6.2 Simulation Setup

For evaluation, we model our SCONNA accelerator from Fig. 6.8 using our developed custom, transaction-level, event-driven python-based simulator (https://github.com/uky-UCAT/SC_ONN_SIM.git). Using the simulator, we simulated the inference four CNN models (with batch size of 1): GoogleNet[162], ResNet50[65], MobileNet_V2

[138], and ShuffleNet_V2 [208]. We evaluate the metrics such as Frames per second (FPS), FPS/W (energy efficiency) and FPS/W/mm² (area efficiency). We also evaluate the impact of PCA error on Top-1 and Top-5 inference accuracy of the CNN models for ImageNet validation dataset [45].

We compared our accelerator with the analog optical accelerators AMM (DEAPCNN [22]) and MAM (HOLYLIGHT [109]) at 8-bits integer quantization for CNN inference. We omitted comparison with CMOS-based digital CNN accelerators as prior analog optical photonic CNN accelerators have outperformed them [22, 157]. We simulate analog optical accelerators for 5 GS/s [94] and from Section 6.3.1, at $B=4$ -bit precision, we set $N=16$ for AMM (DEAPCNN), and $N=22$ for MAM (HOLYLIGHT). Prior works, AMM (DEAPCNN) and MAM (HOLYLIGHT) employ weight stationary dataflow, therefore our evaluation is based on weight stationary dataflow. For fair comparison, we perform area proportionate analysis. In the area proportionate analysis, we altered the VDPE count of each analog optical accelerator, across all of the accelerator’s VDPCs, to match with the area of the SCONNA accelerator having 1024 VDPEs. The scaled VDPE count of MAM (HOLYLIGHT) and AMM (DEAPCNN) are 3971 and 3172, respectively.

Table 6.4 gives the parameters used for evaluating the overhead of the peripherals in our evaluated accelerators. We consider each laser diode to emit input optical power of 10 mW (10 dBm) (Table 6.3)[22], multiplexer and splitter parameters are taken from [109].

6.6.3 Evaluation Results

Fig. 6.9(a) compares the FPS values (log scale) achieved by each accelerator across various CNNs. SCONNA significantly outperforms the analog optical accelerators MAM (HOLYLIGHT) and AMM (DEAPCNN) by $66.5\times$ and $146.4\times$, respectively, on gmean across the CNNs. These benefits are mainly associated with the superior N and higher BR of SCONNA compared to the analog optical accelerators. Because of the high N , SCONNA requires less number of *psums* for DKVs with $S>44$ (refer Table 6.2), while generating the final VDP result. The reduced *psums* drastically reduces the *psum* reduction latency. The higher operating BR=30Gbps compensates for the lengthy stochastic bit-streams of $2^B=256$ bits used by SCONNA. The improvements for SCONNA are more evident for large CNNs such as GoogleNet[162] and ResNet50[65] compared to smaller CNNs such as MobileNet_V2[138] and ShuffleNet_V2[208]. This is due to the fact that MobileNet_V2[138] and ShuffleNet_V2[208] employ depthwise separable convolutions which use DKVs with $S<44$ more frequently than large CNNs. Overall, SCONNA gives exceedingly better FPS compared to the analog optical accelerators.

Fig. 6.9(b) gives the energy efficiency (FPS/W) values for each accelerator across various CNNs. It is evident that SCONNA attains substantially better energy efficiency than the analog optical accelerators. Our SCONNA gains $90\times$ and $183\times$ better FPS/W against analog MAM (HOLYLIGHT) and AMM (DEAPCNN), respectively, on gmean across the CNNs. These energy efficiency benefits are due to the improved throughput and flexible precision support of SCONNA VDPCs. The

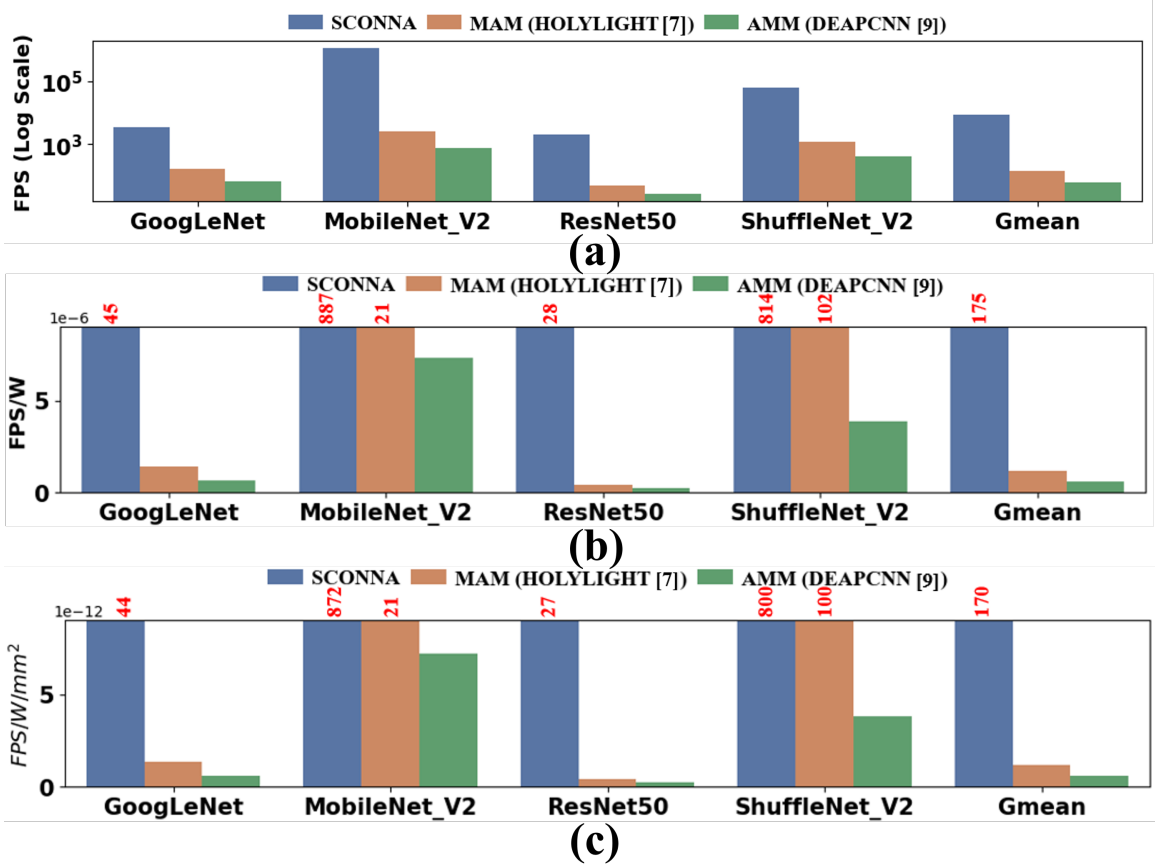


Figure 6.9: (a) FPS (Log Scale) (b) FPS/W (c) FPS/W/mm² for SCONNA versus MAM and AMM accelerators for $B=8$ -bits.

analog MAM (HOLYLIGHT) and AMM (DEAPCNN), due to their limited 4-bit precision, employ two VDPEs to attain an 8-bit precision using bit-slicing. This decreases the throughput and also increases the energy consumption compared to SCONNA VDPCs. In addition, during area proportionate analysis, MAM (HOLYLIGHT) and AMM (DEAPCNN) get scaled to large VDPE counts (3971 and 3172), leading to overall higher static power consumption compared to SCONNA. Therefore, SCONNA achieves better energy efficiency compared to all the other tested accelerators.

Fig. 6.9(c) shows the area efficiency values (FPS/W/mm²) for each accelerator across various CNNs. The area efficiency results look similar to energy efficiency as we match the area of all the accelerators to SCONNA (for the area proportionate analysis). SCONNA gains 91 \times and 184 \times better FPS/W/mm² against analog MAM (HOLYLIGHT) and AMM (DEAPCNN), respectively, on gmean across the CNNs. Overall, SCONNA significantly improves the throughput, energy efficiency and area efficiency compared to the tested analog optical accelerators.

Table 6.5: Top-1 and Top-5 inference accuracy comparison of SCONNA versus MAM for 8-bit quantized CNNs {GoogleNet (GNet), ResNet50 (RNet50), MobileNet_V2 (MNet_V2), ShuffleNet_V2 (SNet_V2)} and ImageNet dataset [45].

SCONNA ACCURACY DROP (%)	GNet [162]	RNet [65]	MNet_V2 [138]	SNet_V2 [208]	Gmean
TOP-1	0.1	0.4	1.5	0.5	0.4
TOP-5	0.1	0.3	0.7	0.4	0.3

6.6.4 Inference Accuracy Results

As discussed in Section 6.4.3, the ADC in the PCA circuits of our SCONNA VDPCs incurs the mean absolute percentage error of 1.3% on the computed binary results. To evaluate the impact of these errors on the CNN inference accuracy, we simulated the inference of four CNNs on SCONNA and analog optical accelerator MAM (HOLY-LIGHT). We integrated our custom simulator with ML-framework PyTorch [126] and performed the inference using ImageNet validation dataset [171] (50k images and 1k classes). Table 6.5 reports the Top-1 and Top-5 inference accuracies obtained for our SCONNA and MAM for four 8-bit integer-quantized CNNs. As evident, SCONNA yields Top-1 and Top-5 accuracy drop of only 0.4% and 0.3%, respectively, on gmean across the tested CNNs. The large CNN models ResNet50[65] and GoogelNet[162] have more tolerance to the errors, and hence, they show minimal to zero drop in accuracy for SCONNA. Furthermore, SCONNA’s accuracy drop can be improved by performing stochastic computing aware training of the CNN models on SCONNA [24]. Our SCONNA accelerator’s significant gains in the FPS, FPS/W, and FPS/W/mm², overshadows the minor drop in the CNN inference accuracy.

6.7 Related Work on Optical CNN 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., [109, 157, 201, 16, 206]). These accelerators employ PIC-based Vector Dot Product Cores (VDPCs) to perform multiple parallel VDP operations. Some accelerators implement digital VDPCs (e.g., [145, 94]), whereas some others employ analog VDPCs (e.g., [109, 157, 22, 166]). Different VDPC implementations employ MRRs (e.g., [109, 22, 157, 111, 51]) or MZIs (e.g., [44, 16, 206]) or both (e.g., [145], [94]). The analog VDPCs can be further classified as incoherent (e.g., [109, 157, 22]) or coherent (e.g., [63, 211, 199, 210, 107, 213]). To set and update the values of the individual input and kernel tensors used for vector dot product operations, the incoherent VDPCs utilize the analog optical signal power, whereas the coherent VDPCs utilize the electrical field amplitude and phase. The coherent VDPCs achieve low inference latency, but they suffer from control complexity, high area overhead, low scalability, low flexibility, high encoding noise, and phase error accumulation issues [119]. In contrast, the MRRs-enabled incoherent VDPCs based accelerators achieve better

scalability and lower footprint, because they use PICs that are based on compact MRRs [22], unlike the coherent VDPCs that use PICs based on bulky MZIs. Various state-of-the-art PIC-based optical CNN accelerators are well discussed in a survey Chapter [43]. Because of the inherent advantages of MRR-enabled incoherent VDPCs, there is impetus to design more energy-efficient and scalable CNN accelerators employing MRR-enabled incoherent VDPCs.

6.8 Summary

To mitigate the very strong scalability versus bit-precision trade-off innately present in analog optical CNN accelerators, we demonstrated a merger of stochastic computing and MRR-based CNN accelerators for the first time in this Chapter. We invented an MRR-based optical stochastic multiplier (OSM) and employed multiple OSMs to forge a novel stochastic computing based CNN accelerator called SCONNA. Our evaluation results for four CNN models show that SCONNA provides improvements of up to $66.5\times$, $90\times$, and $91\times$ in throughput, energy efficiency, and area efficiency, respectively, compared to two analog optical accelerators AMM and MAM, with Top-1 accuracy drop of only up to 0.4% for large CNNs and up to 1.5% for small CNNs.

6.9 Acknowledgments

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

Chapter 7 A Comparative Analysis of Microrings Based Incoherent Photonic GEMM Accelerators

7.1 Introduction

Deep Neural Networks (DNNs) achieve high inference accuracy, which has revolutionized their use in various artificial intelligence tasks, such as image recognition, language translation, and autonomous driving [96, 47]. However, DNNs are computationally intensive because they are typically composed of inherently abundant linear functions such as general matrix-matrix multiplication (GEMM). The need to tackle the rapidly increasing computing demands of the abundant GEMM functions of DNNs has pushed for highly customized hardware GEMM accelerators [20, 161]. Among GEMM accelerators in the literature, silicon-photonic GEMM accelerators have shown great promise to provide unparalleled parallelism, ultra-low latency, and high energy efficiency [109, 58, 22, 34, 157]. A silicon-photonic GEMM accelerator typically consists of multiple Dot Product Units (DPUs) that perform a total of M dot product operations in parallel of N -size each. Several DPU-based optical GEMM accelerators have been proposed in prior works. Among them, the Microring Resonator (MRR)-enabled analog DPU-based accelerators (e.g., [109, 22, 58, 145, 157, 166]) have shown disruptive performance and energy efficiencies, due to the MRRs' compact footprint, low dynamic power consumption, and compatibility with cascaded dense-wavelength-division multiplexing (DWDM).

A typical DPU employs five blocks of optical components to manipulate optical signals in five different ways. (1) A splitting block for splitting (copying) N optical signals in M ways to achieve a fan-out of M per optical signal; (2) An aggregation block for aggregation (multiplexing) of N optical signals per waveguide to achieve a fan-in of N per waveguide; (3) A modulation block for modulation of optical signals to imprint input values onto them; (4) A weighting block for weighting of modulated optical signals to achieve analog input-weight multiplication; (5) A summation block to perform summation of optical signals. Prior accelerators arrange these optical signal manipulation blocks within a DPU in an arbitrary order, resulting in different DPU organizations. Different DPU organizations incur different severity levels of various optical crosstalk effects and signal losses, causing different amounts of optical power penalty across different DPU organizations. This variation in optical power penalty causes different DPU organizations to achieve different values of N (fan-in degree/DPE size) and M (fan-out degree/count of parallel DPEs). This is because the achievable peak values of N and M highly depend on the available optical power budget in the DPU, which in turn is determined by the incurred power penalty in the DPU [152]. It can be intuitively hypothesized that different values of N and M would render different DPU organizations with different levels of processing parallelism at the circuit level, and different magnitudes of throughput and energy-area efficiency at the system level. However, no prior work has tested this hypothesis. We address this shortcoming in this Chapter.

To address these shortcomings, we categorize previous analog optical MRR-based DPUs into three groups based on the organization order of optical signal manipulation blocks. We then conduct a comprehensive circuit-level analysis to assess the impact of the organization on overall losses within a DPU. Leveraging the insights from our analysis, we evaluate the scalability limits of each organization at the circuit level and assess system-level performance, including throughput and energy-area efficiency.

Our key contributions in this Chapter are summarized below.

- We classify the DPU organizations from prior work into three categories, namely ASMW, MASW, and SMWA;
- We analyze and discuss the impact of different DPU organizations on various optical crosstalk effects and signal losses;
- We perform a comparative analysis of the impact of different DPU organizations on the scalability of achievable N and M values at different bit precision values;
- We implement and evaluate ASWM, MASW, and SMWA organizations at the system-level with our in-house simulator, and report the performance in terms of throughput (FPS), energy-efficiency (FPS/W), and area-efficiency (FPS/W/mm²), for the inferences of four CNNs.

7.2 Preliminaries

7.2.1 Processing of CNNs on Hardware Accelerators

In CNNs, the major computational requirement arises from convolutional layers. These layers involve convolution operations that can be converted to General Matrix-Matrix Multiplication (GEMM) operations using the Toeplitz matrix or the im2col transformation [161, 3]. As shown in Fig. 7.1, the input feature map (Fmap) belonging to a convolution layer is unfolded into the matrix \mathbf{I} . The weight filters of the convolution layer are flattened and stacked to form the weight matrix (\mathbf{W}). The GEMM operation between \mathbf{I} and \mathbf{W} gives the resultant output matrix (\mathbf{O}). On conventional CPUs/GPUs, GEMM operations are mapped and executed using basic linear algebra subprograms (BLAS) or Cuda BLAS (cuBLAS) [27, 50]. However, conventional CPUs/GPUs cannot efficiently meet the exponentially growing computational demand of modern CNNs. To meet this demand, both industry and academia have proposed various dedicated GEMM accelerators [81, 157, 22, 94], tailored to process CNNs with better performance and energy efficiency.

7.2.2 Related Work on Optical GEMM Accelerators

To accelerate CNN inferences with low latency and low energy consumption, prior work has demonstrated various GEMM accelerators based on photonic integrated circuits (PICs) (e.g., [109, 157, 22, 44, 186]). Typically, PIC-based GEMM accelerators consist of multiple dot product units (DPUs), and each DPU can perform a GEMM operation on multiple constituent dot product elements (DPEs) as parallel

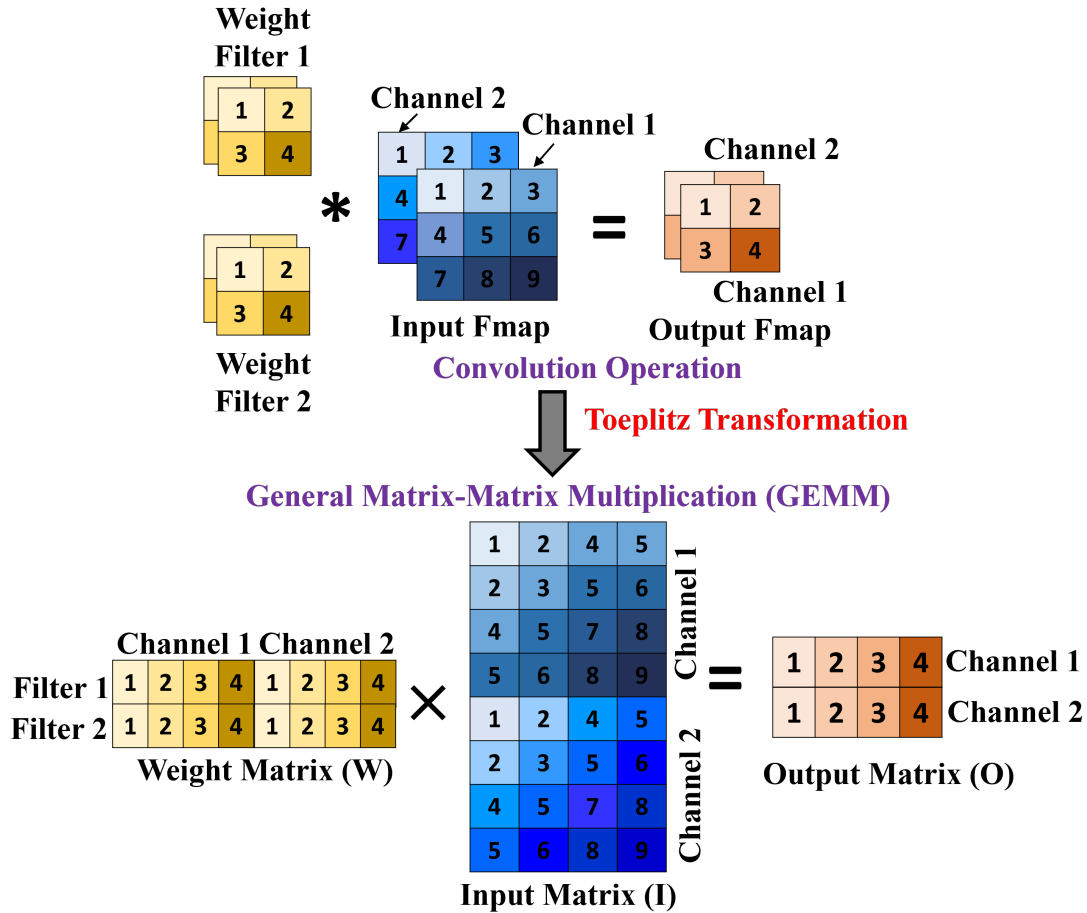


Figure 7.1: Convolution operation at a convolution layer with two weight filters and one input feature map (Fmap) having two channels is transformed into a GEMM operation between input matrix I and weight matrix W .

dot product operations among the rows of the matrix I and the columns of the matrix W . Some accelerators implement digital DPUs (e.g., [145, 94, 186]), whereas some others employ analog DPUs (e.g., [109, 157, 22, 152]). Different DPU implementations employ MRRs (e.g., [109, 22, 157, 111, 186]) or MZIs (e.g., [44, 206]) or both (e.g., [145], [94]). Among these, the accelerators based on MRRs-enabled incoherent DPUs achieve better scalability and lower footprint, because they use PICs that are based on compact MRRs [22], unlike the coherent DPUs that use PICs based on bulky MZIs. Various state-of-the-art PIC-based optical GEMM accelerators are well discussed in survey papers [43, 159, 127].

7.3 Organizations of MRR-based GEMM Accelerators

S. S. Vatsavai et al. in [152] categorized the organizations of optical MRR-based analog DPUs from prior works into two groups: Aggregate-Modulate-Modulate (AMM) and Modulate-Aggregate-Modulate (MAM) [152]. Here, the term "aggregate" refers

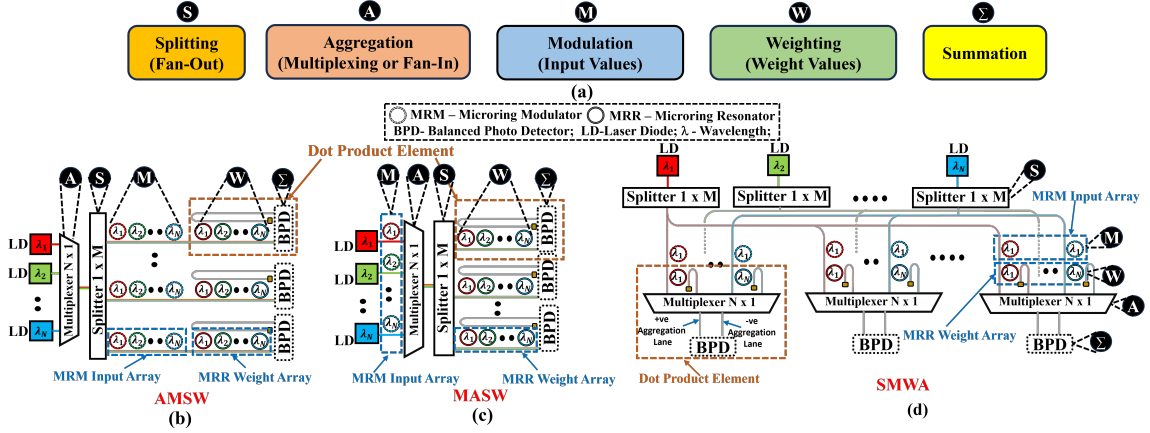


Figure 7.2: (a) Common optical signal manipulation blocks found in optical DPUs. Illustration of common incoherent photonic DPU organizations; (b) AMSW DPU, (c) MASW DPU, and (d) SMWA DPU.

to the aggregation of multiple wavelength channels into a single photonic waveguide through wavelength division-multiplexing (WDM). The first 'Modulate' refers to the modulation of optical wavelength channels with input values, and the second 'Modulate' refers to the modulation (weighting) of input-modulated optical wavelength signals with weight values. This categorization classifies prior MRR-based DPUs into AMM [22, 160, 157] and MAM [109, 201, 9] classes. However, this categorization does not consider the spectrally hitless DPU organization proposed in [34]. In this Chapter, we bridge this gap and also improve the comprehensibility of classification categories based on the order of various optical channel manipulation blocks present in MRR-based GEMM accelerators. The details of these manipulation blocks and their different organizations are discussed next.

7.3.1 Description of Various Blocks that Manipulate Optical Channels in MRR-based GEMM Accelerators

Every DPU in an optical GEMM accelerator employs a total of N laser diodes that generate N optical wavelength channels ($\lambda_1 - \lambda_N$). These optical wavelength channels are manipulated by five different blocks in the DPU, as shown in Fig. 7.2(a). (i) The splitting block (S) splits the optical power of the N optical wavelength channels in M copies, with each copy supporting one DPE. Thus, a total of M DPEs are supported. (ii) Aggregation block (A) multiplexes N wavelength channels into a single waveguide within a DPE to achieve fan-in of N to send N wavelength channels concurrently toward a balanced photodetector (BPD). (iii) Modulation block (M) modulates N or $N \times M$ wavelength channels to imprint a sequence of input values on each of the channels. These input-sequence-imprinted wavelength channels are referred to as optical signals. Each optical signal will thus be a temporal train of optical symbols, with each symbol representing a value from the input sequence. (iv) Weighting block (W)

applies weighting to the input-imprinted wavelength channels (i.e., optical signals). After weighting, each symbol carried in an optical signal represents the product of an input value and a weight value. Therefore, each such weighted optical signal is referred to as an optical product signal, with each temporal symbol of the signal representing a product value. (v) Summation block (Σ) employs incoherent superposition at each BPD to perform symbol-wise summation of the N optical product signals that are sent to the BPD. The BPD generates a resultant photocurrent signal, each temporal symbol of which provides a dot product of size N (i.e., summation of N product symbols/values). This photocurrent signal is fed to a transimpedance amplifier, followed by an analog-to-digital converter, to obtain the sequence of N -sized dot-product results in the digital format [152]. The Σ block involves a total of M BPDs corresponding to M DPEs, thereby generating M parallel dot product results every symbol cycle. In a DPU, generally, the five blocks in the DPU are arranged in a way so that the Σ block comes towards the end of the input-to-output propagation path of optical channels/signals within the DPU, and the \mathbb{M} block always comes before the \mathbb{W} block. However, different DPU organizations can differ in the way the permutational order of the blocks \mathbb{M} , \mathbb{W} , \mathbb{S} , and \mathbb{A} appear in them. Based on this order of these blocks, we classify different DPU organizations found in prior works on MRR-based GEMM accelerators into three categories: ASMW, MASW, and SMWA. Each of these organizations is explained below.

ASMW DPU Organization

Fig. 7.2(b) illustrates a DPU of ASMW organization. First, the \mathbb{A} block appears, wherein N wavelength channels ($\lambda_1 - \lambda_N$) from LDs are multiplexed into a single waveguide. Then, in the \mathbb{S} block, the power of each wavelength channel is equally split into M waveguides to generate a total of $N \times M$ parallel wavelength channels. These wavelength channels then encounter the \mathbb{M} and \mathbb{W} blocks in that order. The \mathbb{M} block (\mathbb{W} block) employs one array of N input MRMs (N weight MRRs) per waveguide. At the output of the \mathbb{W} block, a total of $N \times M$ optical product signals emerge (N signals per waveguide in M parallel waveguides), which are sent to the Σ block containing M parallel BPDs.

MASW DPU Organization

Fig. 7.2(c) illustrates a DPU of MASW organization. Here, the N optical wavelength channels generated by LDs are coupled into N parallel waveguides (one channel per waveguide). Then, the \mathbb{M} block appears wherein each of the N waveguides couples with one input MRM. These N input MRMs in the \mathbb{M} block generate N optical signals, which are then aggregated into a single waveguide in the subsequently appearing \mathbb{A} block. Then, in the \mathbb{S} block, these N optical signals are split into M waveguides to generate their M copies, with each copy feeding a DPE. These $N \times M$ parallel optical signals then encounter the \mathbb{W} block, which employs an array of N weighting MRRs per waveguide. Consequently, at the output of the \mathbb{W} block, a total

Table 7.1: Classification of prior MRR-based analog accelerators based on their DPU organization

DPU Organization	MRR-based DPU Architectures
ASMW	Crosslight[157], DEAPCNN [22], Robin [160]
MASW	Holylight [109], Yang [201], Al-Qadasi[9]
SMWA	Hitless [34]

of $N \times M$ optical product signals emerge, which are sent to the Σ block containing M parallel BPDs.

SMWA DPU Organization

Fig. 7.2(d) illustrates a DPU of SMWA organization. In this organization, the \mathbb{S} block appears first which splits the optical power of each of the N wavelength channels from LDs equally into M separate waveguides by using a total of $N \times M$ splitters. Thus, a total of $N \times M$ wavelength channels and $N \times M$ waveguides emerge, with each waveguide propagating only a single wavelength channel. Then, the \mathbb{M} and \mathbb{W} blocks appear in that order, with both blocks containing a single MRM-MRR pair coupled to each of the $N \times M$ waveguides. At the output of the \mathbb{W} block, a total of $N \times M$ optical product signals emerge (each signal in a dedicated waveguide), which are sent to the \mathbb{A} block. There, a total of M $N \times 1$ multiplexers are used to aggregate the optical product signals in a total of M pairs of aggregation lanes (each pair propagating N optical product signals). These M pairs of aggregation lanes feed the Σ block containing M parallel BPDs. Table 7.1 reports the classification of prior MRR-based GEMM accelerators based on their DPU organization.

7.3.2 Motivation

The performance achieved by the MRR-based DPUs is largely dependent on three parameters (1) The maximum achievable value of N (fan-in degree/DPE size). Often, the achievable value of N for photonic DPUs is less than the dot product size requirement of GEMM operations corresponding to CNN models [152]. In that case, a DPU breaks the dot product into smaller DPU-compatible chunks and generates intermediate results known as partial sums (*psums*). These *psums* are later accumulated using electronic reduction networks [93], to generate the final result. The *psum* reduction latency and energy consumption are non-trivial components of overall latency and energy consumption [120]. Therefore, the value of N plays a crucial role in governing the overall performance of DPUs. (2) The maximum achievable values of M (fan-out degree/count of parallel DPEs). The value of M directly decides the parallelism and consequently achieved throughput by a DPU. (3) The bit precision (B) of input and weight values. If the supported value of B is less than the precision requirement of GEMM operations, bit-slicing is applied to input and weight values [156]. Due to bit-slicing, the overall count of dot product operations increases, degrading the throughput and energy efficiency [186]. Therefore, the fundamental driver

for achieving high performance from optical DPUs lies in maximizing the values of N , M , and B .

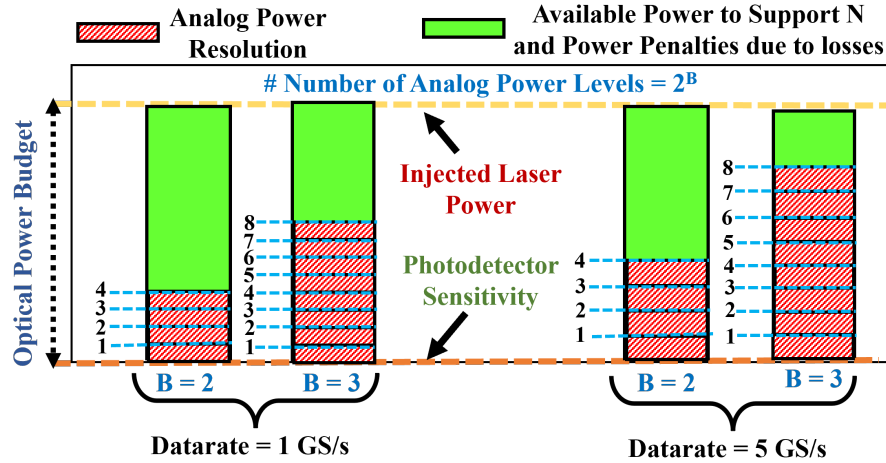


Figure 7.3: Conceptual breakdown of optical power budget usage and dependency of DPU size N on supported bit precision B for different values of $B=\{2, 3\}$ -bits across datarates $DR=\{1, 5\}$ GS/s.

In analog DPUs, a strong trade-off exists among supported values of N , M , and B [9, 152]. The achievable values of M , N , and B also strongly depend on the available optical power budget in the DPUs [9, 152]. This is illustrated in Fig. 7.3, assuming $N=M$, which is a common assumption in the literature [9, 152]. For the bit precision $B=2$, $2^B=4$ analog optical power levels are required that consume a large part of the available power budget, and the remaining power budget is used to support N and power penalty (incurred due to crosstalk effects and signal losses) in the DPU. As B increases to 3-bits, a larger part of the power budget is used to support B , and the available power budget to support N and power penalty further decreases. As a result, the supported value of N decreases too. Similar impact can be observed when the operating datarate of DPUs increases (Fig. 7.3). Low $N=M$ decreases fan-in and fan-out degrees in the DPU, hampering the achievable throughput and energy efficiency. No prior work has characterized this impact, which has motivated this work.

7.4 Circuit-Level Comparative Analysis

In this section, we discuss the impact of DPU organization on power penalties due to various crosstalk effects and optical signal losses.

7.4.1 Impacts on Power Penalty Due to Crosstalk Effects

Inter-modulation crosstalk

From Fig. 7.4(a), inter-modulation crosstalk exists at the **(M)** block. Inter-modulation crosstalk occurs when an MRM unexpectedly modulates a neighboring wavelength channel instead of its assigned wavelength channel [123, 86]. Therefore, it is possible only if there are multiple wavelength channels present in the waveguide at a narrow channel spacing when the MRM is modulating its assigned wavelength channel (Fig. 7.4(b)). Hence, the necessary condition for inter-modulation crosstalk to occur is that the **(M)** block should appear after the **(A)** block in a DPU organization because, then only, the MRMs could have accessible neighboring wavelength channels to unexpectedly modulate them. Therefore, from Table 7.2, inter-modulation crosstalk is present only in the ASMW DPUs.

Cross-weight penalty

The MRR weight arrays in the **(W)** block can exhibit cross-weight penalty [165], as shown in Fig. 7.4(a). Due to an insufficient channel gap between the adjacent optical wavelength channels, a weight MRR could perform undesired weighting on the neighboring optical wavelength channels leading to cross-weight penalty [165]. The additional power required to compensate for this crosstalk noise (to keep the signal-to-noise ratio intact) provides quantification of the cross-weight penalty [165]. Similar to inter-modulation crosstalk, the necessary condition for cross-weight penalty to occur is that the **(W)** block should appear after the **(A)** block in a DPU organization because, then only, the weight MRRs could have accessible neighboring wavelength channels to unexpectedly apply weighting to them. Therefore, from Table 7.2, cross-weight penalty is present only in the ASMW and MASW DPUs.

MRR Filter Penalty

In general, the MRR filter penalty consists of two components [18]: (1) Optical losses due to signal truncation, and (2) inter-channel crosstalk. However, the inter-channel crosstalk component manifests only when an MRR filter is utilized in the demultiplexing configuration to demultiplex a signal from a waveguide containing multiple signals. Since demultiplexing is not required in our considered DPU organizations, the inter-channel crosstalk component remains absent from the MRR filter penalty discussed/analyzed in this Chapter. On the other hand, the optical losses due to signal truncation occur when a modulated optical signal is only partially transmitted through the MRR filter used as a multiplexer [18]. From Fig. 7.4(a), this phenomenon exists at the **(A)** block. Each $N \times 1$ multiplexer consists of multiple MRR filters, and when the passband of a filter does not overlap perfectly with the passband of its corresponding modulated optical signal, it leads to the truncation of the signal side-lobes. Signal truncation is illustrated in Fig. 7.4(c) (on the left); when the filter has a high-quality factor (Q), its passband only partially overlaps with the passband of the modulated optical signal, resulting in signal truncation. Signal truncation occurs

Power Penalty		Losses
Inter-Modulation Crosstalk	M	Propagation Loss S A M W
Signal Truncation Penalty	A	Through Loss M A W
Cross-Weight Penalty	W	Insertion Loss S A M

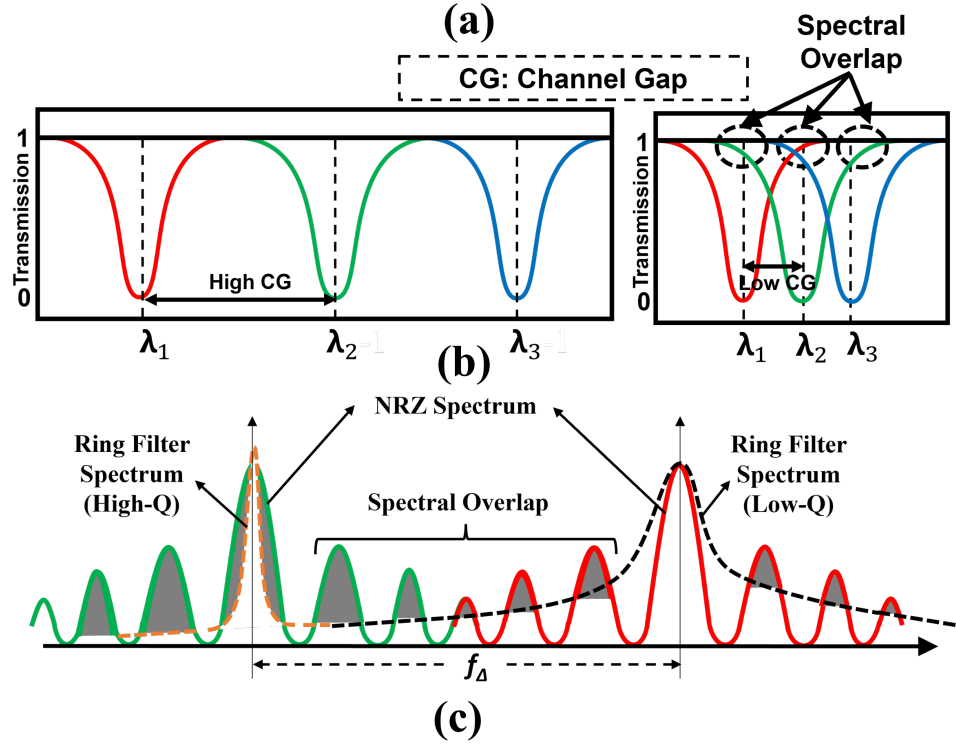


Figure 7.4: (a) Types of losses and power penalties at different optical signal manipulation blocks of optical DPUs. Illustration of (b) Inter-Modulation crosstalk at MRM input arrays [123, 86], and (c) Filter crosstalk and signal truncation at filters [18].

on modulated optical signals only; it does not occur on unmodulated optical wavelength channels. This is because unmodulated wavelength channels do not have any spectral sidelobes. Therefore, the necessary condition for filter penalty to occur is that the **A** block should appear after the **M** block in a DPU organization. The **A** block is organized after the **M** block in both MASW and SMWA DPUs, therefore signal truncation is present in these organizations (Table 7.2).

Table 7.2: Crosstalks effects present in various DPU organizations.

	ASMW	MASW	SMWA
Inter-modulation Crosstalk	✓	X	X
Cross-Weight Penalty	✓	✓	X
Signal Truncation at Filters	X	✓	✓

7.4.2 Impacts Due to Optical Signal Losses

Through losses

The through losses are the optical power losses experienced by a wavelength channel as it traverses through MRMs and MRRs that are out-of-resonance to the wavelength channel in interest but operate on adjacent wavelength channels. From Fig. 7.4(a), through losses are present in blocks **M**, **W**, and **A**. The total amount of through loss experienced by a wavelength channel depends on the number of devices it traverses before reaching BPD. For instance in Fig. 7.2(b), after the splitter, the wavelength λ_1 (indicated with the color red) passes through $(N-1)$ out-of-resonance MRMs and $(N-1)$ out-of-resonance MRRs before reaching the BPD. The reduction in optical power of λ_1 as it interacts with these devices is termed as its through loss. The total through losses can be determined by summing the individual through losses caused by individual MRRs and MRMs. The through losses vary across DPU organizations as reported in Table 7.3. For example, from Fig. 7.2(b), Fig. 7.2(c), and Fig. 7.2(d) the total number out-of-resonance MRMs and MRRs traversed by λ_1 are $2(N-1)$, N , and 2 in ASMW, MASW, and SMWA DPUs respectively. Therefore, λ_1 incurs higher through losses in ASMW DPUs than MASW and SMWA DPU organizations. Similarly, other optical wavelength channels $\lambda_2 - \lambda_N$ also experience higher through losses in ASMW DPUs.

Insertion losses

The insertion losses are the optical power losses encountered by a wavelength channel while devices such as MRMs, MRRs, and filters operate on it. The total insertion losses experienced by wavelength channels are approximately the same across DPU organizations.

Waveguide Propagation Losses

Propagation losses are the sum of scattering losses (due to the sidewall roughness of the waveguide) and absorption losses (due to the material and free-carrier absorption mechanisms in the waveguide). From Fig.7.4(a), propagation losses are present in all the blocks. Propagation losses increase proportionally with the length of the waveguide. The SMWA DPUs, because of their spectrally hitless architecture, employ a larger number of longer waveguides [34], resulting in increased propagation losses compared to ASMW and MASW DPUs. Additionally, MASW DPUs experience lower propagation losses than ASMW DPUs. This is because the MRR weight arrays in

Table 7.3: Optical losses present in various DPU organizations.

	ASMW	MASW	SMWA
Through Losses	High	Moderate	Low
Propagation Losses	Moderate	Low	High

Table 7.4: Definition and values of various parameters used in Eq. 7.1, Eq. 7.2, and Eq. 7.3 (from [9]) for the scalability analysis.

Parameter	Definition	Value
P_{Laser}	Laser Power Intensity	10 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
P_{EC-IL}	Fiber to Chip Coupling Insertion Loss	1.44
$P_{MRR-W-IL}$	Silicon Waveguide Insertion Loss	0.3 dB/mm
$P_{splitter-IL}$	Splitter Insertion Loss	0.01 dB
P_{MRM-IL}	Optical Microring Modulator Insertion Loss	4 dB
P_{MRR-IL}	Optical Microring Resonator Insertion Loss	0.01 dB
$P_{MRM-OBL}$	Out of Band Loss	0.01 dB
$P_{Penalty}$	MASW Network Penalty	4.8 dB
	ASMW Network Penalty	5.8 dB
	SMWA Network Penalty	1.8 dB

MASW DPUs share a single MRM input array, which reduces the overall waveguide length and corresponding propagation losses.

7.4.3 Scalability Analysis

To determine the achievable size N for our ASMW, MASW, and SMWA DPU organizations, we adopt scalability analysis equations (Eq. 7.1, Eq. 7.2, and Eq. 7.3) from [9] and [152]. Table 7.4 reports the definitions of the parameters and their values used in these equations. The reported $P_{Penalty}$ includes the summation of inter-modulation crosstalk, cross-weight penalty, filter penalty, and propagation losses. We consider optimistic values for these parameters, with inter-modulation crosstalk of ≤ 1 dB, cross-weight penalty of ≤ 3 dB, and filter penalty of ≤ 0.5 dB. To achieve such optimistic inter-modulation crosstalk and cross-weight penalties the channel spacing should be equal to $0.4 \times \text{FWHM}$ [165]. We considered Free Spectral Range (FSR=50nm) [9] with FWHM=0.7nm, resulting in channel spacing of 0.25nm(=0.4 \times 0.7). Then, the

FSR limited N value is $200(=FSR/0.25)$. We consider $M=N$ and first solve Eq. 7.1 and Eq. 7.2 for a set of DRs= $\{1, 5, 10\}$ GS/s, to find a corresponding set of P_{PD-opt} . Then, we solve Eq. 7.3 for the maximum value of N that achieves $P_{O/p}$ greater than obtained P_{PD-opt} values across the set of DRs. Fig. 7.5 reports the achievable N of ASMW, MASW, and SMWA DPUs for different bit-precision levels (B) across various DRs. The achievable N value defines the feasible number of MRRs per DPE; thus, this N also defines the maximum size of the dot product that can be generated in our DPU. As evident from Fig.7.5, SMWA can support larger N value compared to ASMW and MASW at all bit-precision levels across different DRs. For instance, SMWA achieves larger $N=83$ for 4-bit precision at 1 GS/s, compared to ASMW and MASW, which achieve $N=36$ and $N=43$, respectively. This is because of SMWA's DPU architecture, as reported in Table 7.2, SMWA significantly reduces crosstalk-related power penalty reducing $P_{penalty}$. This enables SMWA to support larger N compared to ASMW and MASW at the same input laser power.

$$B = \frac{1}{6.02} \left[20 \log_{10} \left(\frac{R \times P_{PD-opt}}{\beta \sqrt{\frac{DR}{\sqrt{2}}}} - 1.76 \right) \right] \quad (7.1)$$

$$\beta = \sqrt{2q(RP_{PD-opt} + I_d) + \frac{4kT}{R_L} + R^2 P_{PD-opt}^2 RIN} + \sqrt{2qI_d + \frac{4kT}{R_L}} \quad (7.2)$$

$$\begin{aligned} P_{O/p}(dBm) = & P_{Laser} - P_{SMF-att} - P_{EC-IL} - P_{Si-att} \times N \times d_{MRR} \\ & - P_{MRM-IL} - (N-1)P_{MRM-OBL} - P_{splitter-IL} \times \log_2(M) \\ & - P_{MRR-W-IL} - (N-1)P_{MRR-W-OBL} - P_{penalty} - 10 \log_{10}(N) \end{aligned} \quad (7.3)$$

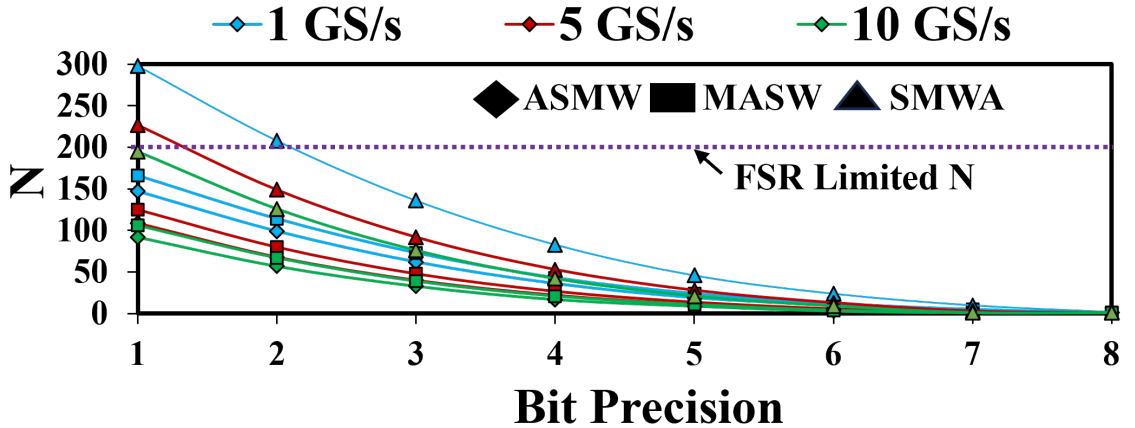


Figure 7.5: Supported DPU size N ($=M$) for bit precision= $\{1, 2, 3, 4, 5, 6, 7, 8\}$ bits at data rates (DRs)= $\{1, 5, 10\}$ GS/s, for AMW, MAW, and MWA DPUs.

Table 7.5: DPU size (N) and DPU Count (#) at 4-bit precision across various DRs for different accelerators architectures.

	Datarate					
	1 GS/s		5 GS/s		10 GS/s	
DPU	N	#	N	#	N	#
ASMW	36	160	17	265	12	291
MASW	43	186	21	275	15	295
SMWA	83	50	42	147	30	198

7.5 Evaluation

7.5.1 System Level Implementation

Fig. 7.6 illustrates the general system-level implementation of incoherent photonic GEMM accelerators. It consists of global memory that stores CNN parameters and a pre-processing and mapping unit. It has a mesh network of tiles. Each tile contains 4 DPUs interconnected (via H-tree) with a unified buffer as well as pooling and activation units. Each DPU consists of multiple DPEs and each DPE is equipped with a dedicated input and output FIFO buffer [192] to store intermittent weights, inputs, and *psums* values. In addition, each tile also contains a *psum* reduction network.

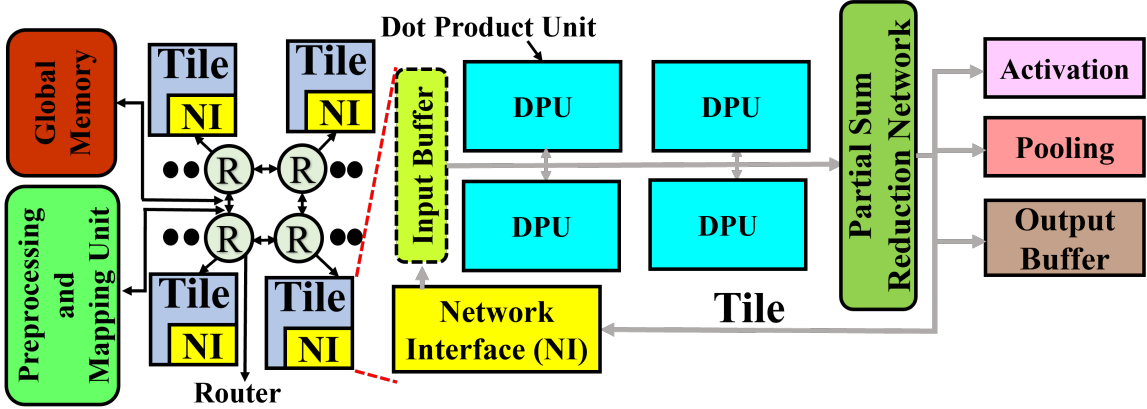


Figure 7.6: System-level overview of Photonic GEMM accelerator.

7.5.2 Simulation Setup

For evaluation, we model system-level implementation of AMSW, MASW, and SMWA GEMM accelerator architectures using our developed custom, transaction-level, event-driven Python-based simulator. Using the simulator, we simulated the inference of four CNN models (with a batch size of 1): GoogleNet[162], ResNet50[65], MobileNet_V2 [138], and ShuffleNet_V2 [208]. We evaluate the metrics such as Frames

Table 7.6: Accelerator Peripherals and DPU Parameters [152]

	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
DAC [181]	12.5	0.78ns	2.50E-3
ADC(1 GS/s) [122]	2.55	0.78ns	2E-3
ADC(5 GS/s) [147]	11	0.78ns	21E-3
ADC(10 GS/s) [60]	30	0.78ns	103E-3
EO Tuning	80 μ W/FSR	20ns	-
TO Tuning	275 mW/FSR	4 μ s	-

per second (FPS), FPS/W (energy efficiency), and FPS/W/mm² (area efficiency). We opted not to evaluate the inference accuracy of the optical GEMM accelerators as prior works [22, 109, 157] indicate minimal or no loss in inference accuracy.

We compared AMSW, MASW, and SMWA accelerator architectures for inference of 8-bit integer quantized CNN models. All accelerators are operated for 4-bit integer precision across data rates 1GS/s, 5GS/s, and 10GS/s, from Fig. 7.5, for these parameters SMWA, ASMW, and MASW achieve N reported in Table 7.5, respectively. We omitted comparison with CMOS-based digital CNN accelerators as prior analog optical photonic CNN accelerators have outperformed them [22, 109, 157]. Our evaluation is based on output stationary dataflow. For a fair comparison, we performed area proportionate analysis, wherein we altered the DPU count for each photonic CNN accelerator across all of the accelerator’s DPUs to match with the area of SMWA ($N=83$) having 50 DPUs. Table 7.5 reports the scaled DPU count of ASMW, MASW, and SMWA across various datarates.

Table 7.6 gives the parameters used for evaluating the overhead of the peripherals in our evaluated accelerators. We consider each laser diode to emit input optical power of 10 mW (10 dBm) (Table 7.4)[22], multiplexer and splitter parameters are taken from [109].

7.5.3 Evaluation Results

Fig. 7.7(a) shows Normalized FPS results for various accelerators with batch size=1 at different datarates (DRs), normalized to ASMW for ResNet50 at 10 GS/s. SMWA accelerator outperforms MASW and ASMW accelerators, respectively, on gmean across four CNN models for all data rates. At 1 GS/s, SMWA achieves up to 2.5 \times and 2.3 \times better FPS than ASMW and MASW, respectively. As DR increases to 5 GS/s and 10 GS/s, SMWA shows better improvements in FPS, achieving up to 3.9 \times and 4.4 \times

better FPS than ASMW, respectively. Similarly, SMWA achieves up to $3.6\times$ and $3.9\times$ better FPS than MASW at 5 GS/s and 10 GS/s.

These significant improvements in throughput for SMWA are due to two reasons. First, inter-modulation crosstalk and cross-weight related power penalties are absent in SMWA (refer Table. 7.2) due to its hitless architecture. This allows SMWA to support a larger DPU size ($N=83$), i.e., the size of the dot product operation N (refer Table 7.5) and the number of parallel dot product operations $M (=N)$. Consequently, the overall throughput is increased with improved parallelism. Secondly, larger N generates less number of *psums* which reduces the use of a partial sum reduction network. This, in turn, reduces the latency associated with *psum* reductions and improves FPS. Among ASMW and MASW, MASW performs slightly better than ASMW at all datarates. MASW with input array sharing mitigates inter-modulation crosstalk power penalty at MRM input array and also incurs lower through losses compared to ASMW (refer Table 7.3), due to these benefits MASW achieves slightly better N compared to ASMW (refer Table 7.5). MASW with higher N achieves better parallelism and decreases reduction latency, resulting in better FPS.

Furthermore, as datarate increases the FPS of each accelerator decreases, at 5GS/s and 10 GS/s, the value of N decreases for all the accelerators (refer Table 7.5) which results in a higher number of *psums*. Therefore, the latency corresponding to *psum* reduction increases with an increase in datarate and leads to lower FPS for accelerators. Overall, SMWA architecture with higher N achieves better throughput than ASMW and MASW architectures.

Fig. 7.7(b) shows FPS/W (log scale) results for ASMW, MASW, and SMWA accelerator with batch size=1 at different DRs, normalized to ASMW for ResNet50 at 10 GS/s. It is evident that the SMWA accelerator attains better energy efficiency than the MASW and ASMW accelerators. At 1 GS/s, SMWA gains $1.9\times$ and $2.5\times$ better FPS/W against analog MASM and ASMW, respectively, on gmean across the CNNs. As the datarate increases to 5 GS/s and 10 GS/s, SWMA achieves $3.17\times$ and $3.3\times$ improvements in FPS/W when compared to MASW. SMWA also exhibits a significant $4.4\times$ and $5\times$ improvement in FPS/W when compared to ASMW at 5 GS/s and 10 GS/s. These energy efficiency benefits of SMWA are due to the improved throughput and decreased energy consumption of *psum* reductions. As discussed earlier, superior N supported by SMWA improves parallelism which decreases dynamic energy consumption with improved throughput. In addition, SMWA also requires the least number of *psum* reductions and this provides energy savings by reducing the usage of *psum* reduction network. At higher datarates of 5 GS/s and 10 GS/s, the N value decreases, consequently requiring more *psum* reductions and *psum* reduction energy consumption. Furthermore, as datarate increases the accelerator peripherals like ADCs consume more static power (refer Table 7.6) which also decreases the FPS/W achieved by each accelerator. Thus, as datarate increase the FPS/W decreases for ASMW, MASW, and SMWA accelerators. Overall, SMWA provides better energy efficiency compared to the other accelerators across different DRs.

Fig. 7.7(c) shows the area efficiency values (FPS/W/mm²) for each accelerator across various CNNs. The area efficiency results look similar to energy efficiency as we match the area of all the accelerators to SMWA (for the area proportionate analysis).

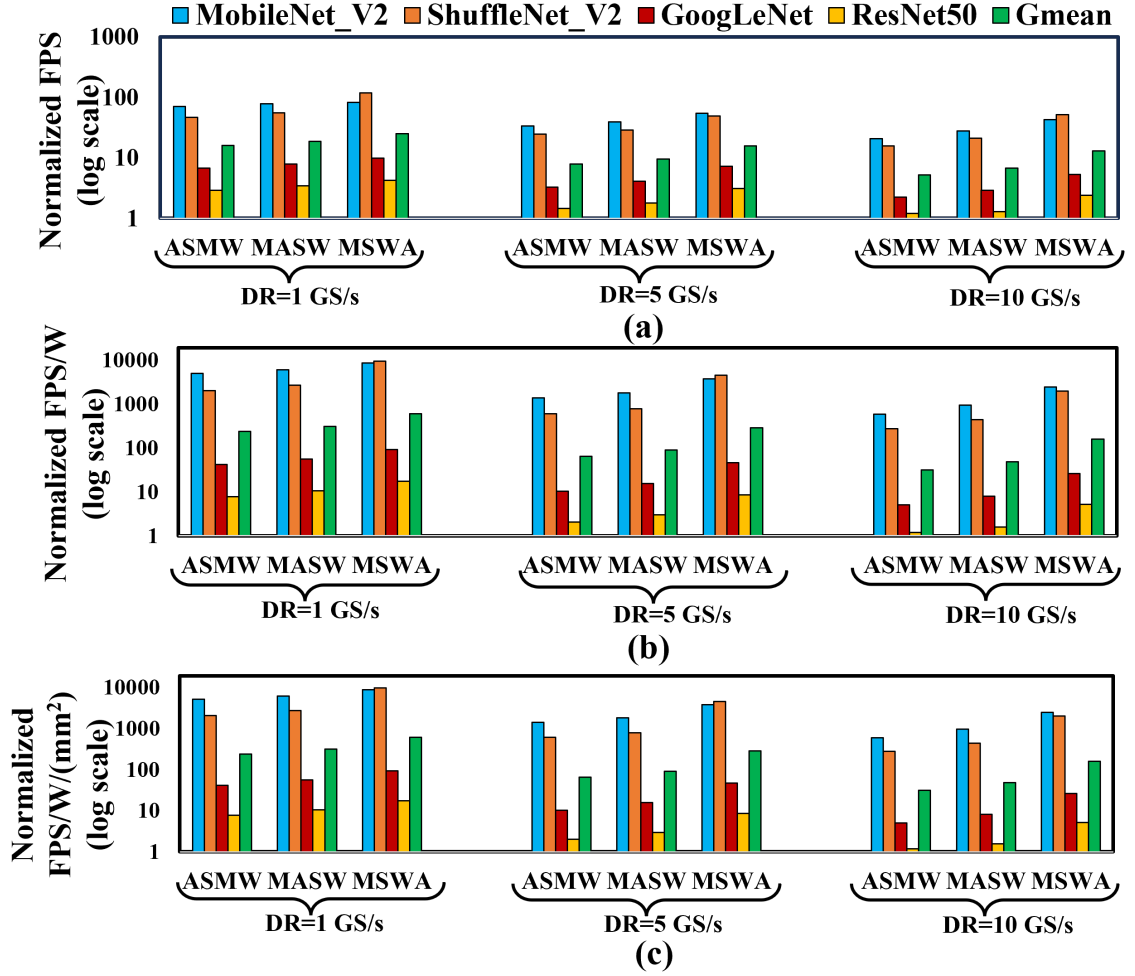


Figure 7.7: (a) Normalized FPS (log scale) (b) Normalized FPS/W (log scale) (c) Normalized FPS/W/mm² (log scale) for AMW, MAW, and MWA accelerators with input batch size=1. Results of FPS, FPS/W, FPS/W/mm² are normalized with respect to AMW executing ResNet50 at 10 GS/s.

SMWA gains up to $5.2 \times$ and $3.4 \times$ better FPS/W/mm² against ASMW and MSAW, respectively, on gmean across four CNN models for all data rates. Overall, the SWMA accelerator achieves better throughput, energy efficiency, and area efficiency compared to the MSAW and ASMW accelerators.

7.6 Summary

In this Chapter, we introduced a systematic approach for classifying prior incoherent MRR-based GEMM accelerators into three distinct categories based on their organization of optical signal manipulation blocks: (1) Modulation-Aggregation-Splitting-

Weighting (MASW), (2) Aggregation-Splitting-Modulation-Weighting (ASMW), and (3) Splitting-Modulation-Weighting-Aggregation (SMWA). We performed a comprehensive circuit-level comparative analysis of MASW, ASMW, and SMWA organizations and identified that each organization incurs different magnitudes of crosstalk noise and optical signal losses. As a result, our scalability analysis at the circuit level demonstrated that each organization achieves different levels of processing parallelism. At the system level, our evaluation results for four CNN models show that SMWA organization achieves up to $4.4\times$, $5\times$, and $5.2\times$ better throughput, energy efficiency, and area-energy efficiency, respectively, compared to ASMW and MASW organizations on average.

Chapter 8 A Hybrid Time-Amplitude Analog Optical Accelerator with Flexible Dataflows for Energy-Efficient CNN Inference

8.1 Introduction

Deep Neural Networks (DNNs) achieve high inference accuracy, which has revolutionized their use in various artificial intelligence tasks, such as image recognition, language translation, and autonomous driving [96, 47]. Convolutional Neural Networks (CNNs) are specific types of DNNs [105]. CNNs are computationally intensive, and hence, require a long inference time. In CNNs, around 80% of the total processing time is taken by convolution operations. The need to tackle the ever-increasing complexity and inference time of CNNs has pushed for highly customized CNN hardware accelerators [20]. For hardware acceleration of convolution operations, they can be converted into general matrix-matrix multiplication (GEMM) operations [161]. The GEMM operations can be further decomposed into dot product operations to be efficiently mapped onto the hardware for acceleration. Often, for efficient and swift hardware-based acceleration, CNNs are quantized to have integer input/weight parameters [91].

Among the CNN hardware accelerators from the literature, silicon-photonic accelerators have shown great promise to provide unparalleled parallelism, ultra-low latency, and high energy efficiency [109, 58, 22, 34, 157]. Typically, a silicon-photonic CNN accelerator consists of multiple Dot Product Units (DPUs) that perform multiple dot product operations in parallel. Several DPU-based optical CNN accelerators have been proposed in prior works based on various silicon-photonic devices, such as Mach Zehnder Interferometer (MZI) (e.g., [206], [44]) and Microring Resonator (MRR) (e.g., [22], [157]).

Among these optical DPU-based CNN accelerators from prior work, the MRR-enabled analog DPU-based accelerators (e.g., [109, 22, 58, 145, 157, 166]) have shown disruptive performance and energy efficiencies, due to the MRRs' compact footprint, low dynamic power consumption, and compatibility with cascaded dense-wavelength-division-multiplexing (DWDM). However, these accelerators face several challenges that hinder their scalability, throughput, and energy efficiency. These prior accelerators employ a combination of microring modulators (MRM) input array and MRR weight bank to perform dot product operation between input and weight values. The inter-modulation crosstalk in MRM input array and inter-spectral, electrical, and thermal crosstalk effects in MRR weight banks reduce the available optical power budget in DPUs. The reduction in the optical power budget significantly reduces the achievable DPU size and supported bit-precision [9, 152]. Moreover, the presence of crosstalk and high spectral sensitivity in MRR weight banks and MRM input arrays needs two separate feedback control units per MRM/MRR [167, 52], one for parameter (input/weight) tuning and one for thermal stability. This increases the static power consumption, diminishing the energy efficiency advantages. Furthermore, none of the prior works have leveraged the ability of balanced photodetectors

(BPDs) as in-situ spatio-temporal accumulators to enable flexible support for various dataflow and to reduce the required buffer accesses and corresponding latency and energy overheads.

To address these shortcomings, this Chapter presents a novel **H**ybrid **T**ime-**A**mplitude **a**nalog optical **A**ccelerator called HEANA. HEANA employs a novel design of hybrid Time-Amplitude Analog Optical Modulator (TAOM) in a spectrally hitless arrangement to eliminate spectral-interference and various crosstalk effects. Our TAOM uses a single active MRR to perform multiplication operations, thereby allowing electro-optic tuning of both input and weight values. Moreover, due to the single-MRR implementation of TAOMs, HEANA achieves a significant reduction in number of active MRR devices compared to prior works. This reduces not only the area consumption but also the insertion losses in HEANA, thereby increasing its energy efficiency. Moreover, HEANA employs Balanced Photo-Charge Accumulators (BPCAs), which inherently support the temporal accumulation of a very high number of partial sums (*psums*) in situ, supporting input stationary, weight stationary, and output stationary dataflows. This eliminates the need for frequent analog to digital conversions and buffer accesses for *psums*, to consequently reduce the overall latency and energy consumption of CNN processing.

Our key contributions in this Chapter are summarized below.

- We present our invented, novel, dataflow-flexible CNN accelerator called HEANA, which employs an array of hybrid time-amplitude analog optical modulators (TAOMs) in spectrally hitless DPU architecture and highly scalable in-situ spatio-temporal accumulators called Balanced Photo-Charge Accumulators (BPCAs);
- We perform detailed modeling and characterization of our invented TAOM and BPCA using photonics foundry-validated, commercial-grade, photonic-electronic design automation tools;
- We perform a comprehensive scalability analysis of our HEANA DPUs, to determine their achievable maximum size (degree of achievable spatial parallelism) and supported bit-precision;
- We implement and evaluate HEANA for input stationary, output stationary, and weight stationary dataflows at the system- level. We compare its performance with two well-known photonic CNN accelerators from prior works for the inference of four state-of-the-art CNNs with two different batch sizes.

8.2 Preliminaries

8.2.1 Processing of CNNs on Hardware

In deep CNNs, the major computation requirement arises from convolutional layers. The convolution operations involved in these layers can be converted to General Matrix-Matrix Multiplications (GEMMs) using a relaxed form of the Toeplitz matrix

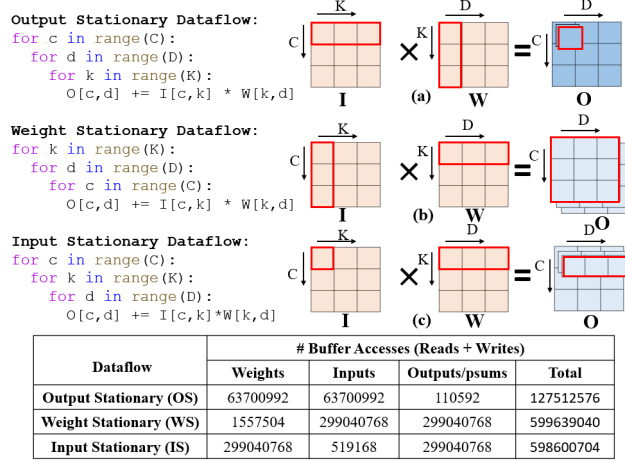


Figure 8.1: Comparison of CNN dataflow schemes: (a) Output Stationary (b) Input Stationary (c) Weight Stationary. Table reports the buffer accesses required by DPU to process layer 5 of GoogleNet[162].

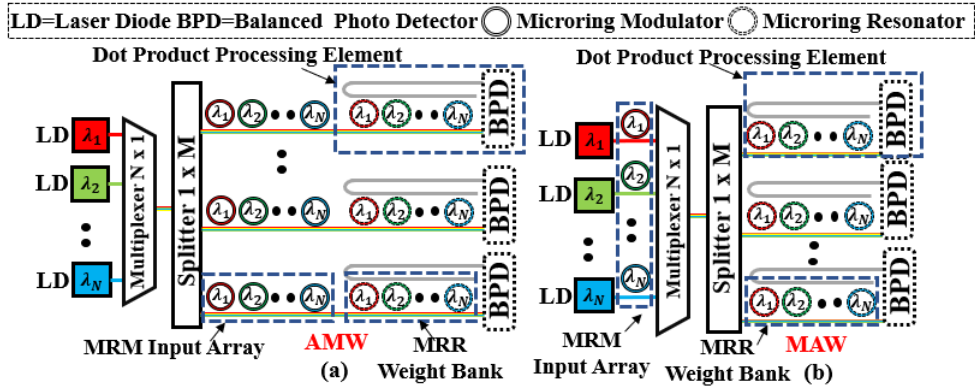


Figure 8.2: Illustration of common analog optical DPU organizations. (a) AMW DPU (b) MAW DPU.

[161]. The Toeplitz matrix \mathbf{I} of the input tensor of a convolution layer can be obtained with the `im2col` method (equivalent to the PyTorch’s `unfold` method) [3]. The weights of the convolution filter tensors are flattened and stacked to form the weight matrix (\mathbf{W}). The GEMM operation between \mathbf{I} and \mathbf{W} gives the resultant output matrix (\mathbf{O}). On conventional CPUs/GPUs, GEMMs are mapped and executed using basic linear algebra subprograms (BLAS) or Cuda BLAS (cuBLAS) [27, 50]. For dedicated CNN hardware accelerators, the mapping and execution dataflow of a GEMM function is determined by the loop ordering used for implementing the function. From Fig. 1(a), a GEMM function requires three nested loops involving the dimensions C , K , and D of the matrices \mathbf{I} and \mathbf{W} . The order of these dimensions in the nested loops determines the frequency of accesses to the \mathbf{I} , \mathbf{W} , and \mathbf{O} matrices. For example, in Fig. 1(b), the given loop order forces the indexes of matrix \mathbf{W} (i.e., k and d) to change less frequently compared to indexes of matrix \mathbf{I} (i.e., c and k) and matrix \mathbf{O}

(i.e., c and d). Therefore, the data values of the matrix \mathbf{W} are accessed in a more stationary manner compared to matrices \mathbf{I} and \mathbf{O} ; hence, this execution is known as weight stationary (WS) dataflow. Similarly, the output stationary (OS) and input stationary (IS) dataflows are illustrated in Fig. 1(a) and Fig. 1(c), respectively.

In CNN accelerators, the data values of matrices \mathbf{I} , \mathbf{W} , and \mathbf{O} are typically read from and written into a unified buffer [93]. The number of accesses to this unified buffer plays a significant role in the overall throughput and energy efficiency achieved by the accelerator. The total number of buffer accesses corresponding to a GEMM function changes based on the choice of dataflow. In Fig. 8.1, the table inset reports the total buffer accesses (reads plus writes) required to finish the execution of the GEMM function corresponding to layer 5 of GoogleNet [162]. The total buffer accesses in the table include accesses to the values of matrices \mathbf{I} , \mathbf{W} , and \mathbf{O} . An output write access is generally performed when the accelerator produces an output value. However, accelerators often produce multiple intermediate results towards a single output value. These intermediate results are known as *psums* (a shorthand for partial sums), which have to be accumulated together (often using a psum reduction network [93, 130, 81]) to calculate the final output value. In case psums are required, the output accesses would involve buffer accesses for both the final output values and *psums*. As shown in the table inset of Fig 8.1, the total buffer accesses vary across the WS , IS , and OS dataflows. The WS and IS dataflows result in the least number of weight and input accesses, respectively, from the unified buffer. Similarly, the OS dataflow results in the least output accesses. Atop dataflows, the total access count also depends on the dimensions C , K , and D of the matrices.

8.2.2 Related Work on Optical CNN Accelerators

Electronic ASICs have traditionally been the preferred choice for implementing CNN accelerators [81]. In light of the deceleration of Moore’s Law, coupled with the exponential growth in CNN complexity [20], electronic ASIC accelerators are struggling to meet the processing speed and energy efficiency demands for large-scale deployment of complex CNN models. To tackle this challenge, both industry and academic researchers are now investigating innovative more-than-Moore technologies that can offer consistently faster and energy-efficient hardware solutions for CNN acceleration in the foreseeable future. Among various technologies, silicon photonics stands out as a promising candidate, offering unparalleled parallelism, ultra-low latency, and high energy efficiency [44, 94, 157]. Silicon photonics-based accelerators harness linear photonic phenomena, such as optical transmission and optical signal superposition within photonic integrated circuits [44, 22], to accelerate CNN inference. This acceleration results in remarkably fast processing speeds and subnanosecond input-to-output latency, following an $O(1)$ scaling law.

To accelerate CNN inferences with low latency and low energy consumption, prior works proposed various accelerators based on photonic integrated circuits (PICs) (e.g., [109, 157, 22, 44, 186]). These accelerators employ PIC-based Dot Product Units (DPUs) to perform multiple parallel dot product operations. Some accelerators implement digital DPUs (e.g., [145, 94, 186]), whereas some others employ analog

DPUs (e.g., [109, 157, 22, 152]). Different DPU implementations employ MRRs (e.g., [109, 22, 157, 111, 186]) or MZIs (e.g., [44, 206]) or both (e.g., [145], [94]). The analog DPUs can be further classified as incoherent (e.g., [109, 157, 22]) or coherent (e.g., [63, 211, 199]). To set and update the values of the individual input and weights used for vector dot product operations, the incoherent DPUs utilize the analog power amplitudes of optical signals, whereas the coherent DPUs utilize the electrical field amplitude and phase. The coherent DPUs achieve low inference latency, but they suffer from control complexity, high area overhead, low scalability, low flexibility, high encoding noise, and phase error accumulation issues [119]. In contrast, the accelerators based on MRRs-enabled incoherent DPUs achieve better scalability and lower footprint, because they use PICs that are based on compact MRRs [22], unlike the coherent DPUs that use PICs based on bulky MZIs. Various state-of-the-art PIC-based optical CNN accelerators are well discussed in survey papers [43, 159, 127]. Because of the inherent advantages of MRR-enabled incoherent DPUs, there is an impetus to design more energy-efficient and scalable CNN accelerators employing MRR-enabled incoherent (analog) DPUs.

Organizations of optical incoherent analog DPUs:

S. S. Vatsavai et al. in [152] categorized the organizations of optical analog DPUs from prior works into two groups: Aggregate Modulate Modulate (AMM) and Modulate Aggregate Modulate (MAM) [152]. Here, 'Aggregate' refers to an aggregation of multiple wavelength signals into a single photonic waveguide through wavelength-division-multiplexing (WDM). The first 'Modulate' refers to the modulation of optical wavelength signals with input values, and the second 'Modulate' refers to the modulation (weighting) of input-modulated optical wavelength signals with weight values. To avoid confusion between two 'Modulate' terms, we propose to replace the second 'Modulate' with 'Weight' to imply the weighting of input-modulated signals. Thus, we categorize DPU organizations as Aggregate Modulate Weight (AMW) and Modulate Aggregate Weight (MAW).

Fig. 8.2 illustrates the schematics of DPU organizations of AMW and MAW categories. In general, these DPUs employ microring modulator (MRM) input arrays and MRR weight banks. In a DPU, inputs are modulated as analog power amplitudes of optical wavelength signals using the MRM input arrays. The individual MRMs of an MRM input array are parallel coupled to a photonic waveguide that carries the multiplexed optical wavelength signals. These optical wavelength signals, after being modulated and multiplexed into the waveguide by the MRM input array, are sent to MRR weight banks. Each MRR in an MRR weight bank is a tunable spectral filter, which consists of an add-drop microring (i.e., a microring coupled to two parallel bus waveguides; one input-through waveguide, and another add-drop waveguide). Each MRR independently controls the transmission of exactly one optical wavelength signal to generate a weighted optical wavelength signal. Each weighted optical wavelength signal is basically a temporal train of optical amplitude symbols carried on an optical wavelength propagating in the waveguide. This amplitude symbol represented the product of the corresponding input and weight values. In a DPU, each waveguide in

fact propagates multi-wavelength weighted optical signals, which are multiplexed by the MRM input array, to a balanced photodetector (BPD) implemented at the end of the waveguide. The BPD Balanced photodetection of these weighted, mutually incoherent WDM signals, through the in Figure 8.2, The BPD produces an electrical current signal, which is a temporal train of electrical current amplitudes. Each current amplitude represents the signed sum of the multi-wavelength temporally coincident optical amplitude symbols that are part of the incident multi-wavelength weighted optical signals [167]. In other words, each current amplitude represents the dot product of the incident wavelength-parallel input and weight vectors, and hence, the electrical current signal at the output of each BPD represents a dot-product signal (i.e., a temporal train of dot-product results). Since a total of M BPDs are employed in a DPU (Fig. 8.2), each DPU generates a total of M parallel dot-product signals. Each symbol of a dot-product signal could be either the final output value or a psum.

8.2.3 Motivation

The CNN accelerators from prior works, which are based on incoherent analog DPUs, have three shortcomings. *First*, the inter-modulation crosstalk in MRM input arrays [152, 88] and inter-spectral, electrical, and thermal crosstalk effects in MRR weight banks [167] put forth a strong trade-off between the achievable DPU size (N) (determined by the achieved wavelength parallelism - Fig. 8.2) and supported bit-precision (B) of AMW and MAW DPUs [9, 152]. This is because the collective power penalty induced by these crosstalk effects can substantially reduce the available optical power budget in DPUs, which can significantly reduce (i) the tolerance to high optical losses caused by large DPU sizes, and (ii) the dynamic range of optical power required to support large bit-precision. As a result, it is shown that MAW DPUs cannot achieve larger than 44×44 size for >4 -bit precision [152]. *Second*, the presence of various crosstalk effects and high spectral sensitivity in the MRR weight banks requires the use of extremely complex control procedures for the actuation of weight values [167, 52]. Such control procedures often employ binary search algorithms [167] or need a feedback control circuit [52]. This requirement increases the implementation complexity, mandating the weight actuation control to be separate from the required thermal stability control per MRR. Thus, each MRR requires two feedback control circuits, one for weight actuation and one for thermal stabilization. Similarly, each MRM already requires a separate input actuation control due to its high-speed operation (typically ≥ 1 gigasymbols per second). This would increase the number of required feedback control units per weighted optical signal to four because both the input MRM and weighting MRR would require one feedback control unit each for thermal stabilization and another unit each for value actuation. Each control circuit consumes a significant amount of static power [52]. As a result, the generation of each N -sized dot-product at a BPD would increase the total static power consumption by $4N \times$, diminishing the overall energy efficiency of the DPUs. *Third*, the AMW and MAW DPUs from prior works fail to fully leverage the ability of BPDs to accumulate *psums* in situ temporally. This ability endows the BPDs with the potential of eliminating the need to use *psum* buffers and dedicated *psum* reduction networks

[93, 130, 94]. Failing to leverage this ability significantly increases the latency and energy consumption of the AMW and MAW DPUs from prior works because of their necessity to frequently read and write *psums* into a buffer and to employ dedicated *psum* reduction networks. Our proposed accelerator, HEANA, tackles these shortcomings, as summarized in Chapter 8.7.

8.3 Our proposed HEANA Architecture

8.3.1 Overview

The main processing unit of our HEANA architecture is a dot product unit (DPU), which is illustrated in Fig. 8.3. A HEANA DPU consists of a comb laser source [88, 193] that emits optical power at a total of N distinct wavelengths (i.e., from λ_1 to λ_N). The N -wavelength (N -chromatic) optical power sourced from the comb laser is split into a total of M waveguide. Each waveguide carries the N -wavelength optical power to a dot product element (DPE).

8.3.2 HEANA’s Dot Product Element (DPE)

From Fig. 8.3, a HEANA DPE employs two artifacts: (i) a spectrally hitless array of a total of N hybrid time-amplitude analog optical modulators (TAOMs), and (ii) a balanced photo-charge accumulator (BPCA). A TAOM generates a weighted optical wavelength signal as a temporal sequence of pulse-width-amplitude-modulated (PWAM) symbols. The total optical energy contained in a PWAM symbol generated by a TAOM represents the analog product of one input and one weight value. The BPCA circuit leverages the temporal charge accumulation and incoherent superposition abilities of photodetectors [29][150] to generate a signed summation of a large number of temporally and spatially arriving PWAM symbols. This signed summation represents the final dot product result, i.e., a value in the final output matrix. The value N here represents the degree of spatial (wavelength) parallelism, which is equal to the number of optical wavelength signals and the number of TAOMs per DPE. The value N is also referred to as the size of the DPE. The arrangements of TAOMs and BPCA, along with their structures and operations, are described in the following subsections.

Spectrally Hitless Array of N TAOMs:

In a DPE, the spectrally hitless array of N TAOMs receives N -wavelength optical power from a waveguide fed from the splitter (Fig. 8.3). On this waveguide, a total of N mono-wavelength filters filter a total of N wavelengths individually, and then drop them onto the respective waveguides of N TAOMs. Thus, each TAOM operates on a unique optical wavelength signal λ_i as discussed in Chapter 8.3.2. This arrangement of TAOMs is spectrally hitless [34] because each TAOM waveguide employs only a single optical wavelength signal. This avoids inter-wavelength interference [17] known as crosstalk from TAOMs operating on adjacent optical wavelength signals.

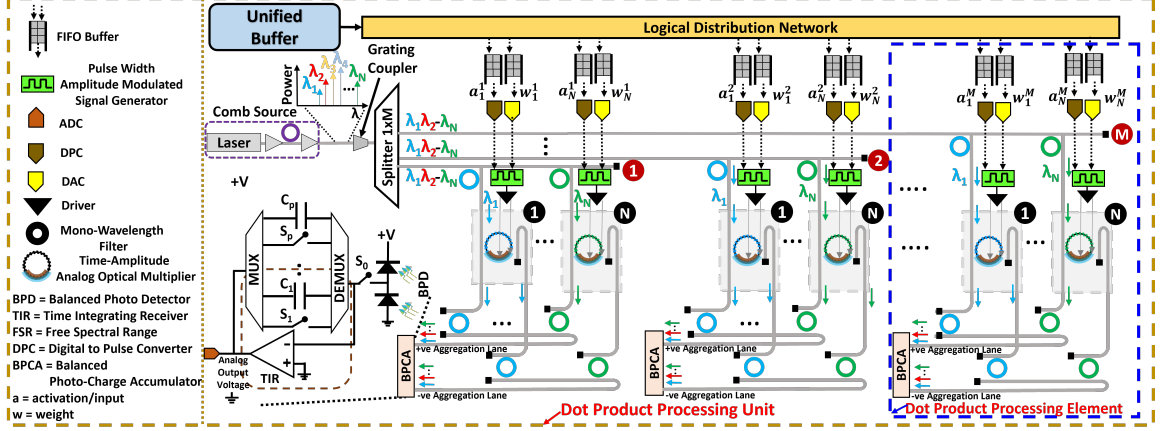


Figure 8.3: Schematic of the Dot Product Unit (DPU) of our HEANA accelerator.

On the contrary, the AMW and MAW DPEs from prior works [109, 22, 152] typically face such crosstalk because they employ a parallel-coupled array of MRMs and MRRs. This arrangement in an N -sized AMW/MAW DPE causes each of the N optical wavelength signals to interact with a total of N MRMs as well as N MRRs, resulting in a substantial amount of crosstalk noise. To minimize this crosstalk noise, AMW/MAW DPEs typically maintain a wide spectral spacing between adjacent optical wavelength signals. However, this restricts the achievable N in the DPEs because a wide wavelength spacing reduces the number of optical wavelength signals that can be spectrally multiplexed within the stringently limited Free Spectral Range (FSR) of the MRMs and MRRs employed in these AMW/MAW DPEs [9, 152]. In contrast, the spectrally hitless arrangement of TAOMs in HEANA completely eliminates the crosstalk noise at TAOMs, which allows for a narrow wavelength spacing to increase the achievable N for HEANA at a given bit precision (more on this in Chapter 8.4).

Each spectrally hitless TAOM receives two operands, i.e., input/activation a_i^M and weight w_i^M , from corresponding FIFO buffers. The activation and weight values are loaded into the FIFO buffers from the unified buffer via a distribution network based on the selected dataflow. Each TAOM is driven by an electrical pulse width amplitude modulated (PWAM) signal. This signal is electro-optically modulated onto the TAOM’s corresponding optical wavelength λ_i to generate an optical PWAM signal. Each PWAM symbol of this signal encodes the input a_i^M as its pulse-width and weight w_i^M as its amplitude so that the total optical energy packetized in the PWAM symbol represents the multiplication result between a_i^M and w_i^M . This multiplication result, depending on its sign, is encoded in the form of a balanced PWAM optical symbol at the through and drop ports of the TAOM. The multiplication results (i.e., balanced PWAM optical symbols) from a total of N TAOMs are then dropped into the positive and negative aggregation lanes via a set of mono-wavelength filters [168] (Fig. 8.3). These aggregation lanes further guide the multiplication results to the BPCA for accumulation. The BPCA, in each symbol cycle, receives N individual multipli-

cation results concurrently (i.e., N PWAM symbols), incoming from N TAOMs on N parallel wavelengths. The BPCA transduces the total optical energy packetized in the N multiplication results (PWAM symbols) arriving at the BPCA into an analog voltage amplitude, which represents a *psum* of the final dot product result (a value in the output matrix \mathbf{O} in Fig. 8.1). Depending on the selected dataflow, a specific capacitor in our BPCA circuit is utilized to transduce and temporarily store the *psum*. The use of capacitors in the BPCA also allows bufferless temporal accumulation of subsequently arriving *psums*. The design and operation of TAOM and BPCA are explained next.

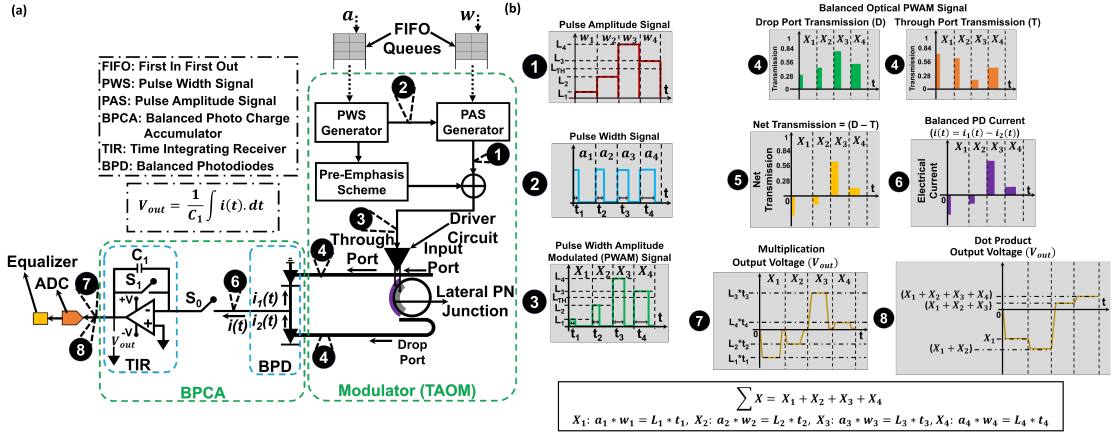


Figure 8.4: (a) Structure of our microring modulator (MRM) based hybrid time-amplitude analog optical modulator (TAOM) connected to a balanced photocharge accumulator (BPCA) and (b) representation of analog signals (optical and electrical) at different stages of TAOM.

Design of Hybrid Time-Amplitude Analog Optical Modulator (TAOM):

Fig. 8.4(a) depicts the schematic of our invented TAOM when it is connected to a balanced photo-charge accumulator (BPCA) unit. As illustrated, our TAOM is basically an add-drop Microring Modulator (MRM) with an embedded lateral PN junction that operates in the forward bias condition. The MRM's peripheral circuitry consists of two queues of FIFO buffers, in which one of them stores the input values (from the input matrix shown in Fig. 8.1) and the other one stores the weight values (from the weight matrix shown in Fig. 8.1), both in the digital binary-radix number format. The FIFO queue for inputs connects to a pulse width signal (PWS) generator and the FIFO queue for weights connects to a pulse amplitude signal (PAS) generator. The output of the PWS is split into two parts: one part is directed to the pre-emphasis scheme, whereas the other part is provided as a reference to the PAS generator. Subsequently, the output of the PAS generator and the output of the pre-emphasis scheme are combined through a current-mode mixer, and the resulting output is a

pulse-width-amplitude-modulated (PWAS) signal. For a complete understanding of the generation of PWAM signals and the underlying circuitry, we direct the readers to [90, 200]. This PWAS signal is routed to a driver circuit. The output of the driver circuit is provided as an electrical bias to the PN junction of the MRM. The output of the MRM (TAOM) is connected to a balanced photo charge accumulator (BPCA) circuit.

Balanced Photocharge Accumulator (BPCA)

Our BPCA circuit is collectively inspired by the time integrating receiver (TIR) design from [150, 1] and the photodetector-based optical pulse accumulator design from [29]. As illustrated in Fig. 8.4(a), a BPCA circuit employs two photodiodes, each connected to the drop and through ports of the MRM. These photodiodes are interlinked in a balanced configuration, commonly referred to as a balanced photodiode (BPD) configuration. The BPD is connected to a TIR via a switch (S_0). The TIR comprises an amplifier and a feedback capacitor/switch (S_1) pair (Fig. 8.4). It functions as a current-to-voltage converter circuit by integrating the incoming electrical current over a period. This ensemble of the BPD and TIR makes the BPCA capable of performing temporal and spatial accumulations (this will be explained in upcoming subsections). Subsequently, the output of the TIR is connected to an analog-to-digital converter (ADC) and an equalizer.

Operation of Integrated TAOM-BPCA Circuit

Figure 8.4(b) illustrates the sequential processing of electrical and optical signals at various stages within our integrated TAOM-BPCA circuit, demonstrating the effective execution of the multiplications and temporal accumulations. As illustrated, the FIFO queue for weight values feeds into the PAS generator, which converts the incoming sequence of digital weight values into a sequence of analog pulse amplitude symbols. This sequence is also called a pulse amplitude signal ((see ❶ in Figs. 8.4(a) and 8.4(b))). Similarly, the FIFO queue for input values feeds the PWS generator, which converts the incoming sequence of digital input values into a sequence of analog pulse-width-modulated symbols. This sequence is called a pulse width signal (see ❷ in Figs. 8.4(a) and 8.4(b)). The PWS output is divided, with one part directed to the pre-emphasis scheme, while the other part serves as a reference for the PAS generator. The output of the PAS generator (current-mode DAC), when mixed with the output of the pre-emphasis scheme, produces a sequence of pulse width amplitude modulated (PWAM) symbols ((see ❸ in Fig. 8.4(b))). This sequence is called PWAM signal. For a complete understanding of the generation of PWAM signals and the underlying circuitry, we direct the readers to [90, 200]. The PWAM signal is fed to a driver circuit, as shown in Fig. 8.4(a). From the driver circuit, the PWAM signal is provided as an electrical input to the PN junction of the MRM.

This input PWAM signal induces free-carrier plasma dispersion in the MRM, which enables the MRM to dynamically adjust the transmission characteristics of the incoming wavelength channel from an external laser source. This action converts the

input electrical PWAM signal into a balanced optical PWAM signal. Here, a balanced optical PWAM signal implies that the original electrical PWAM signal is encoded into optical transmissions simultaneously at both the drop and through ports of the MRM ((See ④ in Fig. 8.4(b))). For each symbol of this balanced optical PWAM signal, the amount of transmission at the through and drop ports of the MRM depends on the amplitude level of each symbol relative to the threshold level in the electrical PWAM signal (L_{TH} in ③ of Fig. 8.4(b)). For instance, the amplitudes of symbols \mathbf{X}_1 and \mathbf{X}_2 in ③ of Fig. 8.4(b) are below the defined threshold level (L_{TH}). Therefore, for symbols \mathbf{X}_1 and \mathbf{X}_2 in ④ of Fig. 8.4(b), the transmission at the drop port of the MRM is lower than the transmission at the through port. As a result, the net transmission, represented as the difference in transmission between the drop and through ports of the MRM (\mathbf{X}_1 and \mathbf{X}_2 in ⑤ of Fig. 8.4(b)), is negative for these symbols. On the other hand, the amplitudes of symbols \mathbf{X}_3 and \mathbf{X}_4 in ③ of Fig. 8.4(b) are above the defined threshold level (L_{TH}). Therefore, for the symbols \mathbf{X}_3 and \mathbf{X}_4 in ④ of Fig. 8.4(b), the transmission at the drop port of the MRM is higher than the transmission at the through port. As a result, the net transmission (\mathbf{X}_3 and \mathbf{X}_4 in ⑤ of Fig. 8.4(b)), is positive for these symbols. Each such symbol of a balanced optical PWAM signal (such as \mathbf{X}_1 in ④ of Fig. 8.4(b)) packetizes certain optical energy that is proportional to the analog product of the corresponding input (a) and weight values (w). For example, the energy packetized in symbol \mathbf{X}_1 represents $\mathbf{a}_1 * \mathbf{w}_1$ (or) $\mathbf{L}_1 * \mathbf{t}_1$ in Fig. 8.4(b). This balanced optical PWAM signal at the output of the TAOM is fed into the BPCA circuit.

Within the BPCA circuit, the BPD transduces the incoming optical pulse sequence ($\mathbf{X}_1, \dots, \mathbf{X}_4$ in ⑤ of Fig. 8.4(b)) from the MRM into a series of differential electrical current amplitudes (⑥ in Fig. 8.4(b)). Here, the differential electrical current amplitude corresponding to each symbol is proportional to the net transmission of the respective optical PWAM symbol (⑤ in Fig. 8.4(b)). Moreover, the multiplication magnitude corresponding to each symbol is encoded as the area under the curve of the differential electrical current symbol. The direction of the electrical current symbol (incoming to the BPD and outgoing from the BPD) represents the sign of the multiplication. This series of differential electrical current symbols is directed towards the TIR via the switch S_0 . The integration of TIR with the BPD introduces a distinctive versatility that enables us to operate our integrated TAOM-BPCA circuit in one of the two distinct modes: (i) multiplier mode or (ii) multiplier and temporal accumulator mode. These modes are explained next.

(i) Multiplier Mode: For this mode, the TIR's sampling speed is matched to the arrival rate of the incoming differential electrical current symbols. At the beginning of each symbol period, opening the switch S_1 allows the electrical current symbol corresponding to that period to linearly charge the feedback capacitor C_1 of the TIR circuit. This linear charging continues for a duration equal to the pulse width of the electrical current symbol. Consequently, the accumulated charge, and therefore, the analog voltage accrued across the feedback capacitor C_1 of the TIR for that symbol period represents the multiplication result related to the corresponding optical PWAM symbol (see ⑦ in Fig. 8.4(b)). Notably, the polarity of the incoming differential electrical current pulse indicates the sign of the multiplication result;

therefore, the polarity of the accrued analog voltage across the TIR's feedback capacitor becomes negative if the incoming electrical current has negative polarity. Once the accrued analog voltage is stable, it is sampled and sent to an analog-to-digital converter (ADC). Then, at the end of the symbol period, closing the switch S_1 allows resetting the charge and voltage on the feedback capacitor to be zero, to prepare the TIR for the next symbol period.

For example, consider the symbol \mathbf{X}_1 in ⑦ of Fig. 8.4(b). The feedback capacitor of the TIR circuit linearly charges until it reaches an analog voltage level of $(L_1 * t_1)$, which represents the signed multiplication result of the symbol \mathbf{X}_1 . After the analog voltage level on the capacitor reaches $(L_1 * t_1)$ for the symbol \mathbf{X}_1 and the capacitor reaches a steady state, the accrued analog voltage is sampled and then sent to the ADC and equalizer for further processing, before the capacitor is made to discharge (reset) by closing the switch S_1 . Here, since the polarity of the differential electrical current corresponding to symbol \mathbf{X}_1 is negative, the accrued analog voltage on the TIR's feedback capacitor is also negative, as shown for symbol \mathbf{X}_1 in ⑦ of Fig. 8.4(b). On the other hand, if the polarity of the incoming differential electrical current pulse is positive, the accrued analog voltage on the TIR's feedback capacitor is also positive, which is illustrated for symbol \mathbf{X}_3 in ⑦ of Fig. 8.4(b). Thus, in this mode, the TAOM-BPCA circuit acts as a multiplier that can produce multiplication results at a fast speed.

(ii) Multiplier and Temporal Accumulator mode: For this case, the TIR's sampling speed is set to be very low compared to the arrival rate of the incoming differential electrical current symbols. Therefore, the series of differential electrical current symbols arriving at the TIR can sequentially charge the TIR's capacitor so that the net accumulated charge and, consequently, the analog voltage accrued on the capacitor over multiple symbol periods provides the signed sum of the individual multiplication results corresponding to different symbols. Thus, this operation essentially performs a temporal accumulation of multiplication results (products). This operation is depicted in ⑧ of Fig. 8.4(b), where the charge accumulates over time based on the incoming electrical current symbols ($\mathbf{X}_1, \dots, \mathbf{X}_4$ in ⑥ of Fig. 8.4(b)), and consequently, the resulting analog voltage accrued on the TIR's capacitor signifies the temporal accumulation result (i.e., $\mathbf{X}_1 + \dots + \mathbf{X}_4$). Moreover, if the incoming differential electrical current pulses to the TIR circuit include both positive and negative polarities, the resultant analog voltage accumulated on the capacitor over time, representing the temporal accumulation operation, is a summation of positive and negative voltages. The temporal accumulation operation illustrated in ⑧ of Fig. 8.4(b) is a summation of both negative and positive voltages. The first two symbols of the incoming current signal (i.e., \mathbf{X}_1 and \mathbf{X}_2 in ⑥ of Fig. 8.4(b)) have negative polarity. Therefore, in ⑧ of Fig. 8.4(b), the net accrued voltage on the capacitor at the end of the second symbol period has negative polarity. The magnitude of this voltage represents $\mathbf{X}_1 + \mathbf{X}_2$. This is because, unlike multiplier mode, the switch S_1 is not closed every symbol period (rather it is kept open) during the operation of this multiplier+temporal accumulator mode. As a result, the accrued voltage does not return to zero at the end of every period; rather, it builds on top of the voltage level accrued in the previous period. In ⑥ of Fig. 8.4(b), the collective magnitude

of the differential electrical current corresponding to the symbols \mathbf{X}_3 and \mathbf{X}_4 is high compared to that of the symbols \mathbf{X}_1 and \mathbf{X}_2 . Therefore, the resultant voltage accrued on the capacitor has positive polarity after all four symbol periods have elapsed. This voltage corresponds to the result of the temporal accumulation of incoming PWAM symbols. Since each PWAM symbol encodes a multiplication result, this temporal accumulation result is basically a temporal dot-product result. This dot-product result can be collected by sampling the accrued voltage and then directing it to the ADC and equalizer for further processing. After the desired number of periods for the temporal accumulations have elapsed, the switch S_1 is closed to reset/discharge the capacitor, to prepare the circuit for the next temporal accumulation.

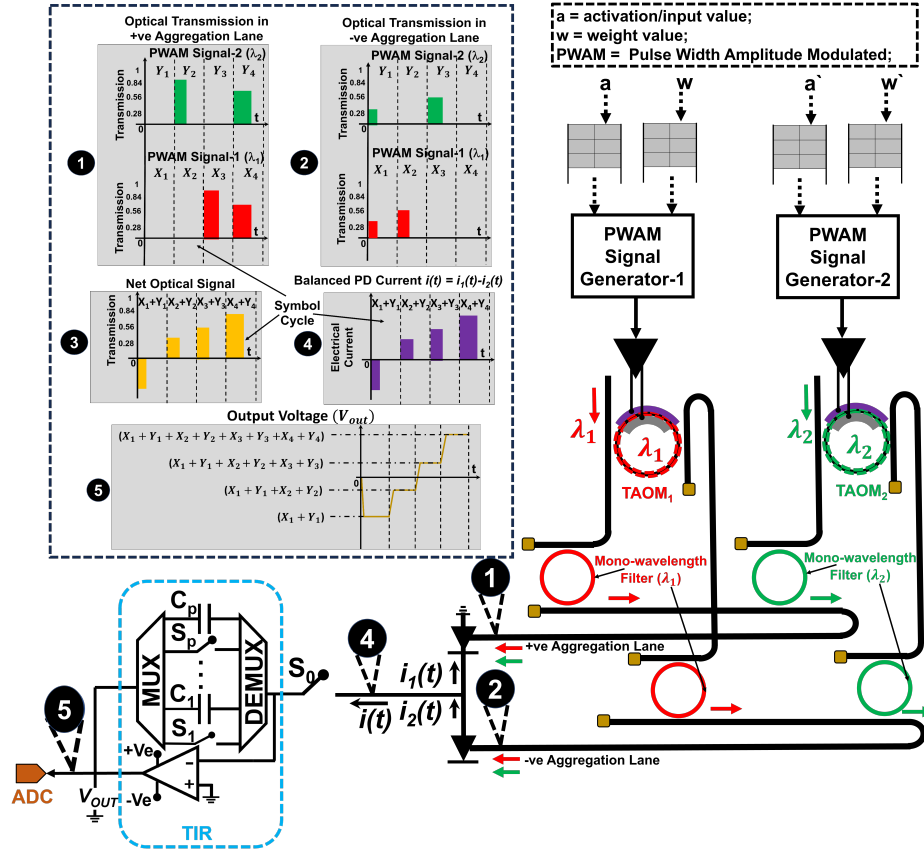


Figure 8.5: HEANA DPE consisting of two spectrally hitless TAOMs, connected to our BPCA circuit. The inset showcases analog representations of signals (both optical and electrical) at various stages of our DPE

Spatio-Temporal Multiply-Accumulate Operations in HEANA DPE:

Previously, we demonstrated that the ensemble of one TAOM and one BPCA can be used to perform temporal multiply-accumulate operations. In this subsection, we extend that idea and demonstrate that an ensemble of multiple TAOMs and a shared BPCA, which creates a HEANA DPE by employing wavelength-division-

multiplexing, can be used to perform spatio-temporal multiply-accumulate operations. To understand this, consider Fig. 8.5 that illustrates the functioning of an example HEANA DPE comprised of an ensemble of two spectrally hitless TAOMs (TAOM₁ and TAOM₂) and a shared BPCA. From what we know about TAOM-BPCA ensemble discussed earlier in Chapter 8.2.4 with respect to Fig. 8.4, TAOM₁ and TAOM₂ in the HEANA DPE of Fig. 8.5 generate balanced optical PWAM signals that are carried onto the dedicated optical wavelengths λ_1 and λ_2 respectively. Both of these balanced optical PWAM signals (shown as optical transmissions in ❶ and ❷ of Fig. 8.5) are multiplexed into the positive and negative accumulation lanes (lanes are waveguides) using mono-wavelength filters. The aggregation lanes guide these balanced optical PWAM signals to the shared BPCA. During each symbol cycle, the BPD of the BPCA performs a balanced incoherent superposition (signed summation) of all the balanced optical PWAM symbols that arrive during the symbol cycle. Consequently, the balanced incoherent superposition first enables the creation of a net optical signal (see ❸ in Fig. 8.5), and then, allows this net optical signal to be transduced to generate a balanced photocurrent signal (see ❹ in Fig. 8.5). The area under the curve of every balanced photocurrent symbol in ❹ of Fig. 8.5 gives a spatial accumulation result (a spatial sum) of the PWAM symbols incident during the corresponding symbol cycle. The polarity of each balanced photocurrent symbol gives the sign of the corresponding spatial sum. Thus, the BPD of the BPCA enables spatial accumulation of incident PWAM symbols. Since all PWAM symbols are multiplication results, their spatial accumulation at the BPCA can also be referred to as spatial multiply-accumulate (MAC) operation or spatial dot-product operation.

This balanced photocurrent signal (which can also be called photocurrent-based spatial MAC signal) produced by the BPD of the BPCA is sent to the TIR of the BPCA, where it can be further processed in two different ways. *First*, if the sampling rate of the TIR is kept equal to the symbol rate of the incoming balanced photocurrent signal, the TIR simply converts the photocurrent-based spatial MAC signal into a voltage-based spatial MAC signal. In this signal, each symbol simply is a voltage level representing a spatial dot-product result. *Second*, if the sampling rate of the TIR is kept to be an integer multiple of the symbol rate of the incoming photocurrent signal, the TIR enables the gradual integration (temporal accumulation) of the individual symbols of the photocurrent-based spatial MAC signal to provide a single voltage level as the final output that represents the temporal sum of all the individual symbols (spatial dot-product results) of the spatial MAC signal. This occurs due to the same operational characteristics of the TIR as discussed in Section 8.2.4-(ii). Thus, the BPCA (BPD + slowly sampled TIR) enables spatio-temporal MAC or dot-product (i.e., temporal accumulations of spatial dot-product results) in the HEANA DPE.

This spatio-temporal accumulation capability of HEANA DPE is clearly illustrated in ❺ of Fig. 8.5. During the first symbol cycle, X_1 and Y_1 are spatially accumulated resulting in an accrued \mathbf{V}_{out} that is proportional to $(\mathbf{X}_1 + \mathbf{Y}_1)$. Here, the cumulative polarity of $(\mathbf{X}_1 + \mathbf{Y}_1)$ is negative, thereby resulting in negative \mathbf{V}_{out} . In the next symbol cycle, the balanced photocurrent generated during the cycle corresponds to spatially accumulated symbols $(\mathbf{X}_2 + \mathbf{Y}_2)$. This balanced photocurrent accrues voltage on top of the \mathbf{V}_{out} accrued in the previous cycle, resulting in updated

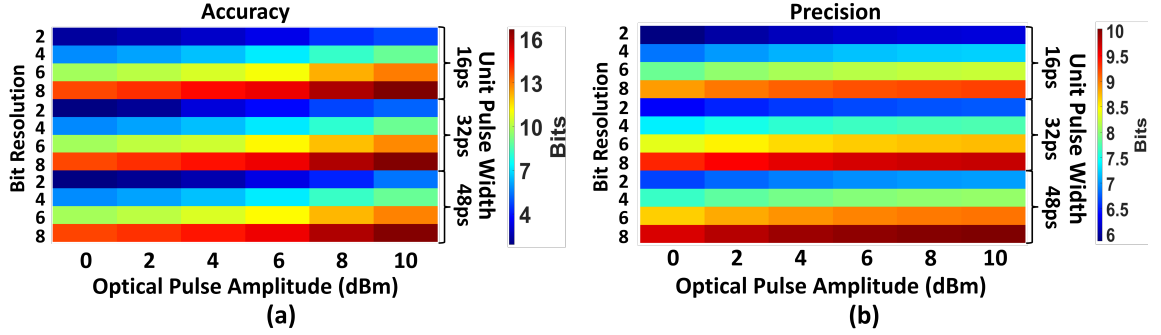


Figure 8.6: Colormap plots that depict the (a) accuracy and (b) precision of our TAOM for different values of input optical power, sample rate and step size/time interval between the time-analog signals

V_{out} that is proportional to $(\mathbf{X}_1 + \mathbf{Y}_1 + \mathbf{X}_2 + \mathbf{Y}_2)$. Thus, the temporal accrual of V_{out} over all four symbol cycles enables the final result to be $(\mathbf{X}_1 + \mathbf{Y}_1 + \mathbf{X}_2 + \mathbf{Y}_2 + \mathbf{X}_3 + \mathbf{Y}_3 + \mathbf{X}_4 + \mathbf{Y}_4)$. The polarity of V_{out} depends on the net polarity of the final result. The final V_{out} from the BPCA circuit is given as input to an ADC to produce the final output in the digital format.

A HEANA DPE can be an ensemble of a shared BPCA and a large number of TAOMs; the number of TAOMs per HEANA DPE is not limited to two. Therefore, this spatio-temporal accumulation capability of HEANA DPE can essentially enable the processing of very large-sized dot-products both spatially as well as temporally. In addition, as shown in Fig. 8.5, the TIR in the BPCA of a HEANA DPE contains a total of p independently operable capacitors. Each HEANA DPE, while leveraging its spatio-temporal accumulation capability, can intermittently switch among these p capacitors by orchestrating the closing and opening of respective switches to facilitate efficient execution of various dataflows. A detailed explanation of this functionality is provided in Section 8.4.

Furthermore, We model our TAOM unit using photonics foundry-validated simulation tools from ANSYS/Lumerical [6]. Here, we perform a time-domain (transient) analysis to evaluate the performance of our TAOM in terms of accuracy and precision for different values of optical power and sample rates. A detailed discussion of this analysis is provided in the next subsection.

Performance Analysis of TAOM:

We evaluate the performance of our TAOM in terms of accuracy and precision. To measure the accuracy of our TAOM, we calculated the logarithmic transformation of the inverse of the normalized mean absolute error (MAE) ($(\log_2(1/(MAE)))$) between the actual voltages (i.e., the voltages across the capacitor (C_1) in the BPCA for different multiplication operations that are extracted from simulations) and the commanded/targeted voltages for different multiplication operations.. We represent

accuracy in terms of bits by considering various input optical pulse amplitudes and bit resolution values. we considered four values of bit resolution (2, 4, 6, 8-bits) and three unit sizes of pulse widths (16ps, 32ps, 48ps). If the unit size of pulse width is 16 ps, the input values of unity would be represented as a 16 ps wide pulse. The colormap plot in Fig. 8.6(a) illustrates the accuracy evaluated for different combinations of optical pulse amplitudes and pulse widths. Additionally, we evaluated precision for these combinations using equations provided in [9]. The corresponding colormap plot for precision can be found in Fig. 8.6(b).

As depicted in the accuracy and precision colormap plots (Fig. 8.6), for a given bit resolution and pulse width for a unity input value, the accuracy and precision of our TAOM increase when the input optical pulse amplitude increases from 0 dBm to 10 dBm. This is because the optical pulse amplitude is basically representative of the optical power signal, and the increase in the input optical pulse amplitude means an increase in the optical power signal. This improves the signal-to-noise ratio, leading to better accuracy and precision for our TAOM. Similarly, for a given input optical pulse amplitude, the precision of our TAOM increases when the pulse width of a unity input value increases. Furthermore, for a given input optical pulse amplitude and a unit pulse width, the accuracy and precision of our TAOM increase with the increase in bit resolution. These results imply that it is possible to achieve accuracy of as high as 16-bit and precision of as high as 10-bit with our TAOM. These accuracy and precision values for our TAOM are highly competitive compared to the values achievable by analog-only photonic incoherent multipliers (or weight banks) from prior works [52][207].

8.4 Mapping of Different Dataflows on HEANA DPU

The mapping and execution dataflow of a GEMM operation is determined by the loop order, as discussed in Section 8.2.1. A GEMM operation requires three nested loops involving the dimensions C , K , D of the matrices \mathbf{I} and \mathbf{W} (Section 8.2.1). The GEMM operation between matrices \mathbf{I} and \mathbf{W} basically requires a total of $C \times D$ K -sized dot products. Fig. 8.7 illustrates the loops used for mapping the GEMM operation between \mathbf{I} and \mathbf{W} onto a HEANA DPU consisting of M DPEs, where each DPE can perform a dot product operation of size N . Here, M also determines the number of parallel dot product operations that can be implemented on the DPU. The supported size N of the DPEs is often less than the required dot product size K for implementing GEMM-converted convolutions [152]. In the wake of such size mismatch between the hardware and mapped matrices, to make the required dot products amenable to hardware implementation, the input and weight matrices (\mathbf{I} and \mathbf{W}) are mapped to the hardware in smaller blocks by tiling them. The M and N parameters of the DPU along with the execution dataflow (loop order) determine the shape of the input and weight tiles. These tiles are mapped both temporally and spatially onto the DPEs to compute blocks of output values as output tiles and facilitate the completion of the GEMM operation. In Fig. 8.7, the innermost loop of each dataflow illustrates how the tile shapes are determined for the input, weight, and output tiles. For the output stationary dataflow illustrated in Fig. 8.7(a), the

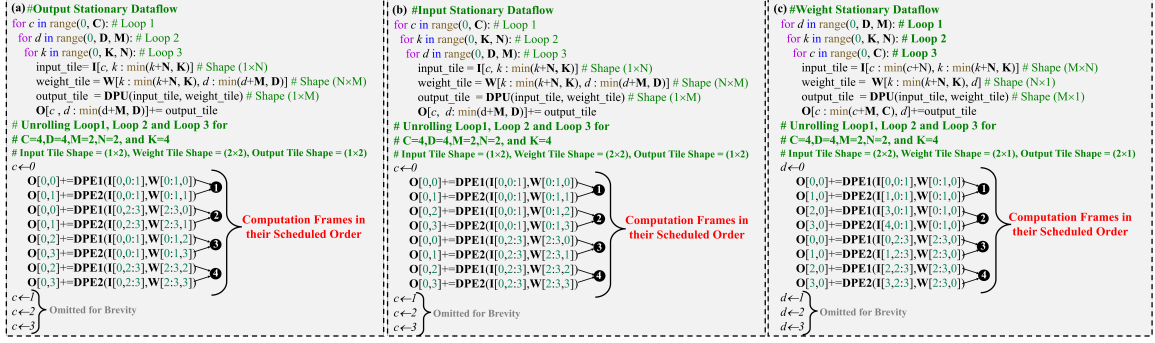


Figure 8.7: Loops involved in GEMM operation between input I and weight W , mapped onto a photonic DPU, with (a) Output Stationary Dataflow (b) Input Stationary Dataflow (c) Weight Stationary Dataflow. DPU consists of 2 DPEs ($M=2$), and each DPE can perform dot product operation of size $N=2$. For output stationary dataflow and input stationary dataflow the loops are unrolling to show the computations involved in the evaluation of output O row 1 whereas for weight stationary dataflow evaluation of output O column 1 is shown. Note that each DPU call performs two dot product operations simultaneously employing the two DPEs. Scheduling order of computation frames for the last three iterations of outermost loops are omitted for brevity

input tile shape is $1 \times N$, whereas for the weight stationary dataflow shown in Fig. 8.7(c), the input tile shape is $M \times N$. Similarly, the shape of the weight and output tiles vary between the output stationary and weight stationary dataflows. Thus, the choice of execution dataflow has an impact on the tiling of I , W , and O matrices of the GEMM operation.

The choice of execution dataflow (i.e., loop order) for a given M and N parameters also determines the computation order for output tiles. In Fig. 8.7, for each dataflow, the loops are unrolled for DPU with parameters $M=2$ and $N=2$ for input matrix I (with dimensions $C \times K=4 \times 4$) and weight matrix W (with dimensions $K \times D=4 \times 4$). After unrolling, each iteration of the outermost loop involves four computation frames (1, 2, 3, and 4). A computation frame is comprised of dot product operations that can be mapped onto a single DPU in a parallel manner. The number of computation frames involved in each iteration of an outermost loop can be derived using $\#ComputationFrames = (\text{ceil}(D/M) \times \text{ceil}(K/N))$. For each computation frame, the input and weight tiles are mapped onto a single DPU to perform parallel dot product operations toward computing a single output tile. The DPU employs two DPEs (DPE1 and DPE2) to perform two dot product operations in parallel. In Fig. 8.7(a), for computation frame 1, the output stationary dataflow computes partial output values of $O[0,0]$ at DPE1 and $O[0,1]$ at DPE2. These output values ($O[0,0]$ and $O[0,1]$) belong to row 1 of output matrix O . In contrast, computation frame 1 of weight stationary dataflow shown in Fig. 8.7(c) computes output values $O[0,0]$ at DPE1 and $O[1,0]$ at DPE2, which belong to the column 1 of the matrix O . Fur-

thermore, among the output stationary and input stationary dataflows, an identical computation frame can involve the computation of different output values belonging to the same output row of the matrix \mathbf{O} . For instance, in Fig. 8.7(a) and Fig. 8.7(b), computation frame ② for both the output stationary and input stationary dataflows compute the output values corresponding to the row 1 of matrix \mathbf{O} ; however, the output stationary dataflow computes the output values $\mathbf{O}[0,0]$ and $\mathbf{O}[0,1]$, whereas the input stationary dataflow computes different output values $\mathbf{O}[0,2]$ and $\mathbf{O}[0,3]$ belonging to the same row. Thus, the computation order of output tiles varies with the choice of execution dataflow. This emphasizes the critical role of the dataflow in shaping the execution order of the GEMM operation.

In the next three subsections, we will present a detailed explanation and a comprehensive comparison of GEMM operations’ mapping onto HEANA and AMW [22] architectures. We will consider output stationary (*OS*), input stationary (*IS*), and weight stationary (*WS*) dataflows. **Important Definitions:** To help with our explanation, we introduce two tiling directives: temporal switching (*ts*) and temporal folding (*tf*) to elucidate the temporal mapping of input and weight tiles. Both the temporal switching and temporal folding of input/weight tiles mean that the input/weight tiles that are mapped onto the DPU change over time in a pipelined manner. The unit amount of time for which the input/weight tiles are held in the DPU for processing before folding (switching) in a pipelined manner is referred to as a folding (switching) cycle. Temporal switching and temporal folding, however, differ in the way that they impact the mapping and processing of the output tile. Temporal switching of the input/weight tile alters the output tile being computed at each switching cycle. In contrast, employing temporal folding means that the computation of the same output tile is folded across multiple cycles, with each folding period generating partial results for each value in the output tile. The lengths of the switching and folding cycles may differ across different dataflows as well as between input and output tiles.

Example DPU and GEMM configurations for explanation of dataflow mappings: For our explanation of various dataflow mappings in the following subsections, we assume the availability of a single DPU composed of 2 DPEs ($M=2$), where each DPE has a size of $N=2$. Each DPE is equipped with one BPCA containing 2 capacitors ($p=2$) to facilitate in-situ spatio-temporal accumulations (as discussed earlier). Furthermore, We consider GEMM operation between input matrix \mathbf{I} (with dimensions $C \times K=4 \times 4$) and weight matrix \mathbf{W} (with dimensions $K \times D=4 \times 4$) to generate output matrix \mathbf{O} (with dimensions $C \times D=4 \times 4$).

8.4.1 Output Stationary Dataflow

Fig. 8.8(a) illustrates the *OS* dataflow mapping. The tiling of the inputs and weights allows the computing of row 1 (\mathbf{O}_1) of the matrix \mathbf{O} . Evaluating row 1 of the matrix \mathbf{O} involves dot product operations between row 1 of the matrix \mathbf{I} and all the columns of the matrix \mathbf{W} . These dot products are broken down into multiple computation frames for scheduling/mapping on the DPU. The scheduling order of the computation frames is defined in Fig. 8.7(a). To map the computation frames, each DPU offers spatial

parallelism at two levels, in general, one at the wavelength level and the other at the DPE level. The parameter N governs the wavelength-level parallelism, while M governs the DPE-level parallelism. For the input matrix \mathbf{I} , the *OS* dataflow exploits the wavelength-level parallelism in the DPEs by tiling the inputs according to $N = 2$ so that the input tile of shape $1 \times N = 1 \times 2$ is spatially mapped on the wavelengths λ_1 and λ_2 of the DPEs. The same input tile is broadcasted to both the DPEs for spatial reuse of the input values. For the weight matrix \mathbf{W} , both the wavelength-level and DPE-level parallelisms are leveraged, and according to the parameters $N=2$ and $M=2$, the shape of the weight tile is determined as $N \times M = 2 \times 2$. Consequently, weights are spatially mapped to wavelengths as well as DPEs. The weight tile is unicast to two DPEs, with each DPE receiving different weight values while using the shared input values to compute different output values. To compute the output row 1 (\mathbf{O}_1), the input and weight tiles undergo temporal switching and folding as shown in Fig. 8.8(a), to evaluate computation frames (①, ②, ③, and ④). According to the computation order of *OS* dataflow, the mapped output tiles are changed fewer times compared to the input and weight tiles.

To understand this mapping, consider the example illustrated in Fig. 8.8(a). In the figure, for computation frame ①, an input tile (yellow tile) is broadcasted whereas the weight tile (yellow tile) is unicast to DPEs. Therefore, DPE1 and DPE2 both receive inputs (i_1^1, i_1^2) while DPE1 receives weights (w_1^1, w_2^1) and DPE2 receives weights (w_1^2, w_2^2) . For computation frame ①, DPE1 and DPE2 compute partial sum (*psum*) results towards output values o_1^1 and o_1^2 , respectively. Then, computation frame changes from ① \rightarrow ②. For computation frame ②, the temporal folding ($tf_1 \rightarrow tf_2$) of input and weight tiles from yellow to pink tiles takes place as shown in Fig. 8.8(a). The updated input and weight tiles (pink tiles) are broadcasted and unicast to DPEs, respectively. Subsequently, the DPE1 and DPE2 generate the remaining *psum* results $(i_1^3 w_3^1 + i_1^4 w_4^1)$ and $(i_1^3 w_3^2 + i_1^4 w_4^2)$ towards output values o_1^1 and o_1^2 , respectively. Thus, the output tile being computed remains constant while the input and weight tiles are changed to evaluate the *psum* results of o_1^1 and o_1^2 . To obtain the final values of o_1^1 and o_1^2 , there is a need to accumulate the *psum* results generated in DPE1 and DPE2 via computation frames ① and ②.

After accumulation, the temporal switching of weights ($tsw_1^1 \rightarrow tsw_1^2$) along with the resetting of the temporal folding to tf_1 is carried out to implement the computation frame ③. This switching of weights moves the weight tile to columns 3 and 4 of the matrix W , thus resulting in the changing of the output tile to o_1^3 and o_1^4 . Computation frame ③ evaluates *psum* results for the newly changed output tile. After completion of computation frame ③, the temporal folding ($tf_1 \rightarrow tf_2$) of input and weight tiles from yellow to pink tiles is again carried out to evaluate computation frame ④. Computation frame ④ generates the residual *psum* results for o_1^3 and o_1^4 . The accumulation of *psum* results from computation frame ③ and computation frame ④ generates the final values of o_1^3 and o_1^4 . This accumulation is performed in HEANA temporally by leveraging the temporal accumulation capability of the BPCA-based design (as explained in Sections 3.2.4 and 3.2.5). With this accumulation, all the values of output row 1 are evaluated. In Fig. 8.8(a), the arrows connecting the tiles illustrate the direction of tile switching (changing), and the count of arrows between

tiles signifies the number of switchings (changes). Thus, from Fig. 8.8(a), it is evident that for the *OS* dataflow the output tiles are switched (changed) the least number of times compared to the input or weight tiles.

Here, it's important to note that the *psum* results generated by DPEs for evaluating computation frames are in analog form, requiring subsequent analog-to-digital conversion. HEANA and prior optical accelerators differ in the required number of analog-to-digital conversions and the nature of the involved *psum* reduction (*psum* accumulation) process. This is discussed next.

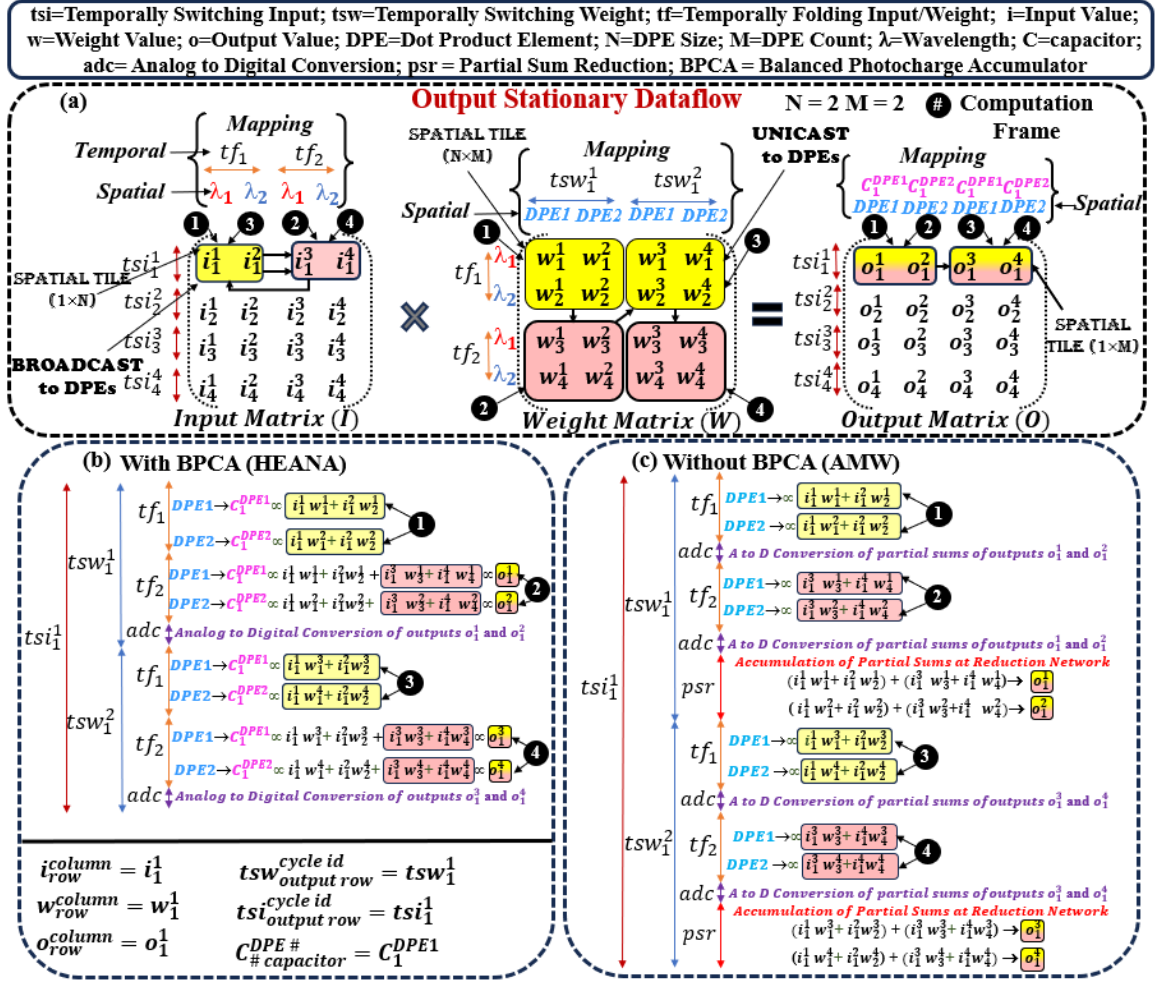


Figure 8.8: Mapping of GEMM operation between I and W with output stationary dataflow on HEANA DPU compared to AMW DPU. (a) Temporal and Spatial tiling of I , W , and O depending on DPU parameters M and N (b) Evaluation of output row 1 (O_1) by HEANA DPU with BPCA (c) Evaluation of output row 1 (O_1) by AMW DPU without BPCA.

Fig. 8.8(b) and Fig. 8.8(c) illustrate the processing of output row 1 O_1 on HEANA and AMW architectures, respectively. As described in Section 8.3.2, HEANA is integrated with BPCA consisting of capacitors that enable spatio-temporal in-situ ac-

cumulations. During temporal switching cycle tsi_1 , two temporal switching cycles of weights (tsw_1^1 and tsw_1^2) take place. Within tsw_1^1 cycle, the DPU calculates output values o_1^1 and o_1^2 involving computation frames ❶ and ❷. As explained above, for the computation frame ❶, during tf_1 , DPE1 and DPE2 generate analog *psums* ($i_1^1 w_1^1 + i_1^2 w_2^1$) and ($i_1^1 w_1^2 + i_1^2 w_2^2$) towards o_1^1 and o_1^2 , respectively. In HEANA, analog *psum* values are stored in analog format as proportional voltage levels on capacitors C_1^{DPE1} and C_1^{DPE2} of BPCAs. Subsequently, for frame ❷, during tf_2 cycle, weight values (w_3^1, w_4^1) are mapped to DPE1, and weight values (w_3^2, w_4^2) are mapped to DPE2. Both DPE1 and DPE2 receive input values (i_3^1, i_4^1) to generate residual *psum* results towards output values o_1^1 and o_1^2 . HEANA accrues the generated residual *psum* results by incrementing a proportionate analog voltage on capacitors C_1^{DPE1} and C_1^{DPE2} atop the voltage level stored during the tf_1 cycle. The final voltage accrued on capacitors C_1^{DPE1} and C_1^{DPE2} is proportional to the final output values o_1^1 and o_1^2 . Thus, HEANA employs in-situ temporal accumulation of *psums*. Finally, the analog to digital conversion of o_1^1 and o_1^2 is carried out to produce final results as shown in Fig. 8.8(b). *In contrast*, the AMW architecture lacks BPCAs, and therefore converts the analog *psum* results generated during tf_1 and tf_2 to digital format using ADCs, before storing these results in buffers. Then, it employs an electronic reduction network such as S_Tree [93], to accumulate the *psum* results from tf_1 and tf_2 to generate the final values for outputs o_1^1 and o_1^2 . Thus, AMW requires more analog-to-digital conversions and needs an external reduction network for performing *psum* accumulations. Similarly, tsw_1^2 computes o_1^3 and o_1^4 involving ❸ and ❹, next tsi_1 gets updated to tsi_2 for generating output pixels of output row 2 O_2 .

In summary, HEANA BPCAs offer several advantages. First, it significantly reduces the required count of analog-to-digital conversions, as *psum* results are intermittently stored in BPCA capacitors. This reduction in analog-to-digital conversion count translates into lower power consumption. Second, HEANA leverages the in-situ temporal accumulation capabilities of BPCA, which eliminates the requirement of a reduction network altogether. This elimination leads to a substantial reduction in *psum* reduction latency and the associated energy consumption.

8.4.2 Input Stationary Dataflow

Fig. 8.9(a) illustrates the *IS* dataflow mapping. The input and weight matrices are tiled similar to the tiling for the *OS* dataflow. However, the *IS* dataflow maps with the least number of changes in input tiles. As a result, as shown in Fig. 8.7(b), the computing functions performed during various computation frames differ between the *IS* and *OS* dataflows. For computation frame ❶, in Fig. 8.9(a), the input tile (yellow tile) is broadcasted, while the weight tile (yellow tile) is unicast to DPEs. DPE1 and DPE2 compute the *psum* results toward output pixels o_1^1 and o_1^2 , respectively. Then, keeping the input tile (yellow tile) unchanged, temporal switching of the weight tile is carried out ($tsw_1^1 \rightarrow tsw_1^2$) to evaluate computation frame ❷, resulting in the generation of *psum* results belonging to o_1^3 and o_1^4 at DPE1 and DPE2, respectively. After completion of computation frame ❷, all computation frames involving the input tile (yellow tile) are completed. Subsequently, temporal switching of weight tile

is again carried out ($tsw_1^2 \rightarrow tsw_1^3$) (labels are not depicted in Fig. 8.9(a)) along with temporal folding ($tf_1 \rightarrow tf_2$) to perform computation frame ③ for generating the residual *psum* results belonging to output values o_1^1 and o_1^2 . Finally, temporal switching of weight tile is carried out ($tsw_1^3 \rightarrow tsw_1^4$) (labels are not depicted in Fig. 8.9(a)) to compute computation frame ④, generating residual *psum* results belonging to output pixels o_1^3 and o_1^4 . In the *IS* dataflow, as evident from the number of arrows between the tiles shown in Fig. 8.9(a), the weight and output tiles are changed more often than the input tiles.

Fig.8.9(b) and Fig.8.9(c) illustrate the processing of output row 1 (O_1) on HEANA and AMW architectures, respectively. In this *IS* dataflow, the order of temporal weight switching cycle (*tsw*) and temporal folding cycle (*tf*) differ from the *OS* dataflow. Initially, tsw_1^1 and tf_1 cycles are scheduled to perform computations of frame ① at DPE1 and DPE2, generating *psum* results toward o_1^1 and o_1^2 which involve inputs i_1^1 and i_1^2 . This *psum* results are generated in the analog format and they are stored as the accrued analog voltage levels on the capacitors C_1^{DPE1} and C_1^{DPE2} using the BPCAs, as from Fig. 8.9(b). Then, by keeping i_1^1 and i_1^2 stationary, a temporal switching of weights to cycle tsw_1^2 is carried to schedule computation frame ② to generate *psum* results belonging to o_1^3 and o_1^4 . In this case, HEANA stores the generated *psum* results in different capacitors, namely C_2^{DPE1} and C_2^{DPE2} , within the BPCAs. This selection of a different set of capacitors is necessary as the *psum* results generated during cycle tsw_1^1 for frame ① and cycle tsw_1^2 for frame ② belong to different output tiles, and therefore they cannot be accumulated on the same capacitors. This is the reason why our BPCAs incorporate multiple feedback capacitors; multiple capacitors enable the spatio-temporal accumulation and storage of multiple *psum* results corresponding to different output tiles to meet the requirement of different dataflow.

During cycles tsw_1^2 and tf_1 , all computations involving i_1^1 and i_1^2 are completed. Following this, cycle tsw_1^3 is scheduled in conjunction with cycle tf_2 , aimed at mapping inputs i_1^3 and i_1^4 for the computation of frame ③ which generates residual *psum* results associated with o_1^1 and o_1^2 . In this process, HEANA utilizes capacitors C_1^{DPE1} and C_1^{DPE2} to store *psum* results generated during cycles tsw_1^1 and tf_1 . The generated *psum* results for frame ③ are temporally accumulated on top of the *psum* results stored on the capacitors during frame ①. At the end, the total voltage accrual provides final output values o_1^1 and o_1^2 , which are converted using analog-to-digital conversion. In contrast to HEANA, in Fig. 8.9(c), the AMW architecture needs multiple analog-to-digital conversions and the utilization of a reduction network to reduce the *psums* associated with o_1^1 and o_1^2 .

Likewise, cycles tsw_1^4 and tf_2 are utilized to compute frame ④ for generating o_1^3 and o_1^4 . For that, HEANA utilizes capacitors C_2^{DPE1} and C_2^{DPE2} for in-situ temporal accumulations of *psums*, whereas AMW once again relies on a reduction network for the accumulation of *psums*.

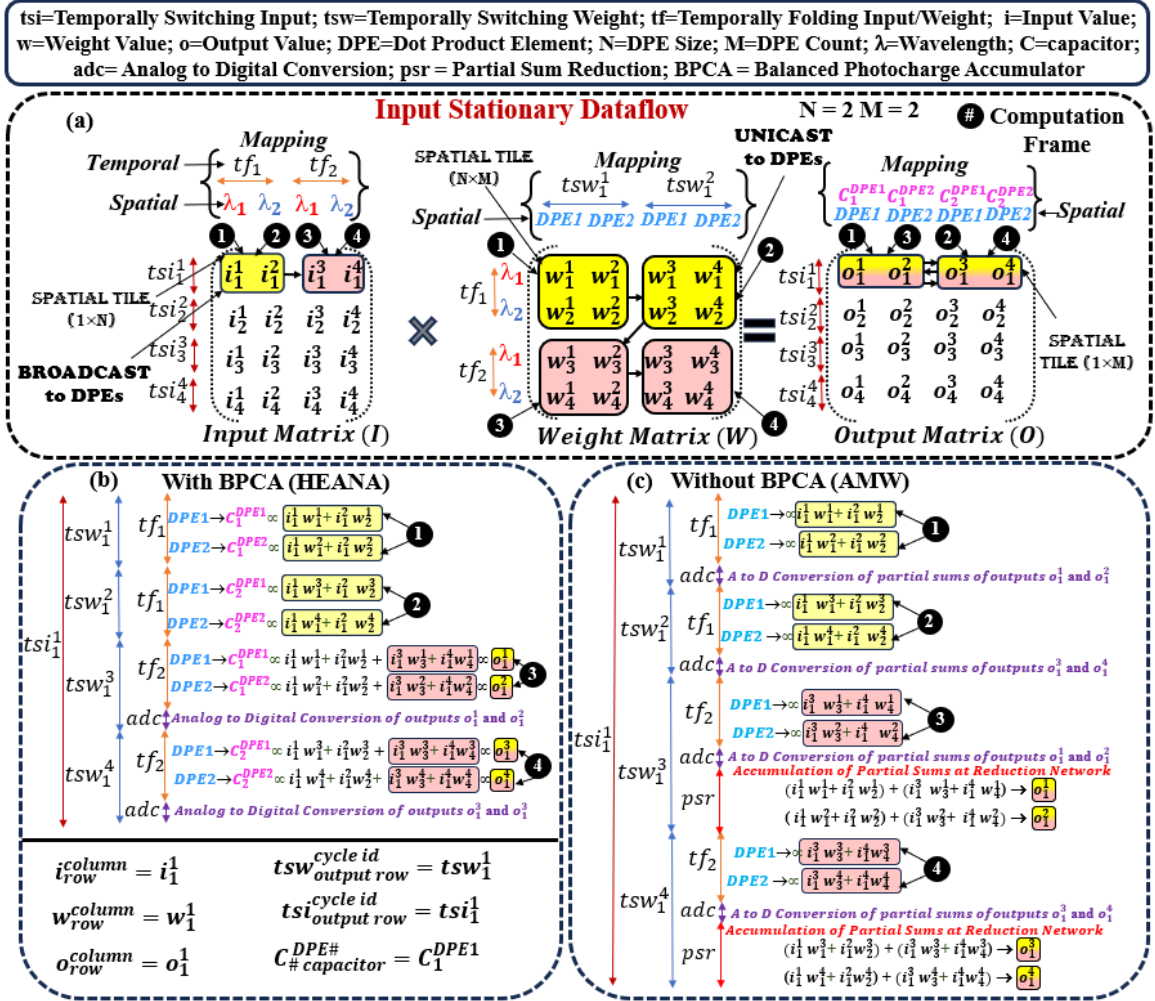


Figure 8.9: Mapping of GEMM operation between I and W with input stationary dataflow on HEANA DPU compared to AMW DPU. (a) Temporal and Spatial tiling of I , W , and O depending on DPU parameters M and N (b) Evaluation of output row 1 (O_1) by HEANA DPU with BPCA (c) Evaluation of output row 1 (O_1) by AMW DPU without BPCA.

8.4.3 Weight Stationary Dataflow

Fig. 8.10(a) depicts the WS dataflow mapping and tiling while computing column 1 (O^1) of output matrix O . Unlike the IS and OS tiling, in the WS dataflow, the inputs are spatially mapped onto both wavelengths and DPEs, while weights are spatially mapped to wavelengths and reused across DPEs. As shown in Fig. 8.10, the shape of the weight tile depends on N , and the input tile shape is determined based on N and M . In the WS dataflow, the required counts of temporal switching for input and output tiles are more compared to the required count of switching of weight tiles. Therefore, the computation order of the tiles in the WS dataflow differs

when compared to the *IS* and *OS* dataflows. In the *WS* dataflow, for computation frame ① (see Fig. 8.10(a)), the input tile (yellow tile) is unicasted, while the weight tile (yellow tile) is broadcasted to DPEs. While processing computation frame ①, DPE1 and DPE2 compute the *psum* results belonging to output values o_1^1 and o_2^1 , respectively. Next, to process computation frame ②, temporal switching of input tile is carried out ($tsi_1^1 \rightarrow tsi_1^2$) to evaluate *psum* results belonging to o_3^1 and o_4^1 at DPE1 and DPE2, respectively. During frame ②, all computing functions involving weight tile (yellow tile) are completed. Subsequently, temporal switching of the input tile is again carried out ($tsi_1^2 \rightarrow tsi_1^3$) (labels are not depicted in Fig. 8.10(a)) along with temporal folding ($tf_1 \rightarrow tf_2$) to change the yellow tiles to the pink tiles. Here, DPE1 and DPE2 process computation frame ③ for generating the residual *psum* results toward output values o_1^1 and o_2^1 . Finally, temporal switching of input tile is carried out ($tsi_1^3 \rightarrow tsi_1^4$) (labels are not depicted in Fig. 8.10(a)) to process computation frame ④ to generate residual *psum* results toward output values o_3^1 and o_4^1 . As evident from Fig. 8.10(a), the input and output tiles are changed more often than the weight tiles in the *WS* dataflow.

Fig. 8.10(b) and Fig. 8.10(c) illustrate the processing of output column 1 O_1 on HEANA and AMW architectures employing the *WS* dataflow, respectively. The *WS* dataflow uses temporal input switching (*tsi*) and temporal folding (*tf*). The tsi_1^1 and tf_1 cycles are scheduled for processing computation frame ①, to generate *psum* results toward o_1^1 and o_2^1 . This involves weights w_1^1 and w_2^1 . The generated *psum* results are stored in analog format on capacitors C_1^{DPE1} and C_1^{DPE2} by HEANA (Fig. 8.10(c)). Subsequently, by keeping weights w_1^1 and w_2^1 stationary, a temporal switching to cycle tsi_1^2 is executed for computation frame ② to generate *psum* results toward output values o_3^1 and o_4^1 . To store these *psum* results, capacitors C_2^{DPE1} and C_2^{DPE2} within the BPCAs are used by HEANA. After cycle tsi_1^2 , all computations involving w_1^1 and w_2^1 are completed. Then, computation frame ③ is executed with cycles tsi_1^3 and tf_2 by mapping weights w_3^1 and w_4^1 , to generate the residual *psum* results associated with output values o_1^1 and o_2^1 . HEANA performs in-situ temporal accumulation of *psum* results generated during computation frames ① and ③ by using capacitors C_1^{DPE1} and C_1^{DPE2} . This accumulation process results in the final voltage levels on the capacitors that are proportional to the final output values o_1^1 and o_2^1 . On the other hand, from Fig. 8.10(c), AMW architecture derives final output values o_1^1 and o_2^1 using multiple analog-to-digital conversions and partial sum reduction at the reduction network. Similarly, to map computation frame ④, the combination of cycles tsi_1^4 and tf_2 is utilized to generate output values o_3^1 and o_4^1 . Here, HEANA utilizes capacitors C_2^{DPE1} and C_2^{DPE2} for in-situ temporal accumulations of *psum* results corresponding to o_3^1 and o_4^1 , whereas AMW once again relies on a reduction network for the accumulation of *psum* results.

As evident from the examples shown in Fig. 8.8, Fig. 8.9, and Fig. 8.10, HEANA requires fewer analog-to-digital conversions to generate an output value. Table 8.1 reports the number of analog-to-digital conversions needed by a DPU to compute all the output values of a GEMM operation. From the table, the AMW and MAW architectures require $\text{ceil}(K/N) \times$ more conversions than HEANA. Hence, HEANA can significantly reduce the energy and latency of analog-to-digital conversions as

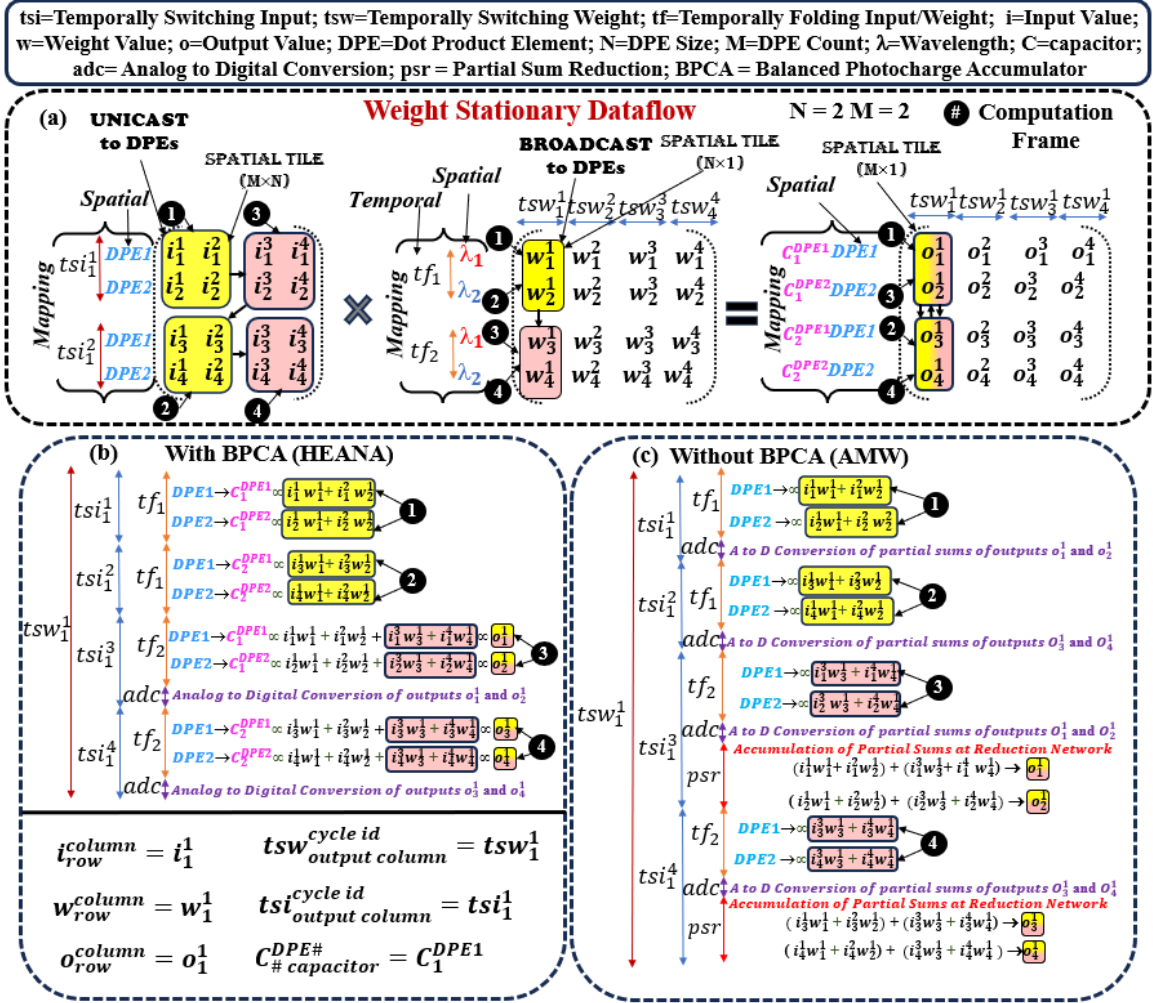


Figure 8.10: Mapping of GEMM operation between I and W with weight stationary dataflow on HEANA DPU compared to AMW DPU. (a) Temporal and Spatial tiling of I , W , and O depending on DPU parameters M and N (b) Evaluation of output column 1 (o_1^1) by HEANA DPU with BPCA (c) Evaluation of output column 1 (o_1^1) by AMW DPU without BPCA.

shown in our system-level analysis (see Section 8.6.3). Furthermore, it is worth noting that HEANA is not only more efficient in terms of energy consumption and latency but also more flexible and compatible with a wide range of input dimensions.

8.4.4 Operational Logistics and Benefits of HEANA

Capacitor Selection Based on Dataflows: For the OS dataflow, the $psum$ results generated by the BPCA in consecutively scheduled computation frames belong to the same output value. Therefore, the same capacitor is selected for several continuous frames until the accumulation of all the $psum$ results belonging to the same output

Table 8.1: Number of analog-to-digital (AtoD) conversions required by various optical DPUs for a GEMM operation. C and D are the height and width of the output matrix, where K is the width (height) of the input (weight) matrix. N=DPE size.

DPU	Dataflow	AtoD Conversions
HEANA	OS	$C \times D$
	WS	$C \times D$
	IS	$C \times D$
AMW/MAW	OS	$C \times D \times \text{ceil}(K/N)$
	WS	$C \times D \times \text{ceil}(K/N)$
	IS	$C \times D \times \text{ceil}(K/N)$

value has been completed (Fig. 8.8(b)). In contrast, for the *IS* dataflow (Fig. 8.9(b)) and the *WS* dataflow (Fig. 8.10(b)), the *psum* results generated by the BPCA in consecutive frames belong to different output values. Therefore, a different capacitor is selected for every frame by controlling the corresponding switches (Fig. 8.3) using a demux. From the sizes of the Toeplitz matrices of the state-of-the-art CNNs [37], we have determined that the BPCA in our HEANA architecture may require up to $p=4608$ capacitors to support seamless accumulation of *psum* results during the execution of the *IS* and *WS* dataflow.

Accumulation Benefits of HEANA: In HEANA, a *psum* result is accumulated as a voltage level accrued on a capacitor of a BPCA. After each frame, the capacitor can hold on to the accrued voltage for a significant amount of time, thus eliminating the need to convert the *psum* result in the digital format via ADC and store it in a buffer temporarily. This eliminates the latency and energy overheads related to ADC conversions and digital buffering (read+write) of *psum* results. Several of such *psum* results might be required to be accumulated together to produce one value in the output matrix \mathbf{O} (Fig. 8.1). The use of TIR in our BPCA enables the accumulation of several such *psum* results temporally over multiple frames (over continuously, intermittently, or sporadically scheduled frames, depending on the utilized dataflow), by allowing a linear increment of the accrued voltage on the capacitor proportion to the *psum* result arriving at each frame. This eliminates the need to employ dedicated *psum* reduction networks, thereby eliminating related latency and energy overheads as well.

8.5 Scalability Analysis

To determine the achievable size N for our HEANA DPU, we adopt the scalability analysis equations (Eq. 8.1, Eq. 8.2, and Eq. 8.3) from [9] and [152, 186]. Table 8.2 reports the definitions of the parameters and their values used in these equations. We consider $M=N$ and first solve Eq. 8.1 and Eq. 8.2 for a set of DRs= $\{1, 5, 10\}$ GS/s, to find a corresponding set of P_{PD-opt} . Then, we solve Eq. 8.3 for the maximum value of N that achieves $P_{O/p}$ greater than obtained P_{PD-opt} values across the set

of DRs . Fig. 7.5 reports the achievable N of HEANA, AMW and MAW DPUs for different bit-precision levels (B) across various DRs . The achievable N value defines the feasible number of multipliers per DPE; thus, this N also defines the maximum size of the spatial dot product that can be generated in our DPU. As evident from Fig.7.5, our HEANA can support larger N value compared to AMW and MAW at all bit-precision levels across different DRs . For instance, HEANA achieves larger $N=83$ for 4-bit precision at 1 GS/s, compared to AMW and MAW, which achieve $N=36$ and $N=43$, respectively. This is due to HEANA's spectrally hitless DPE architecture (as discussed earlier in Section 8.3.2) due to which HEANA significantly reduces the crosstalk-related power penalty contributing to $P_{penalty}$. This enables HEANA to support larger N compared to AMW and MAW at the same input laser power.

$$B = \frac{1}{6.02} \left[20 \log_{10} \left(\frac{R \times P_{PD-opt}}{\beta \sqrt{\frac{DR}{\sqrt{2}}}} - 1.76 \right) \right] \quad (8.1)$$

$$\beta = \sqrt{2q(RP_{PD-opt} + I_d) + \frac{4kT}{R_L} + R^2 P_{PD-opt}^2 RIN} + \sqrt{2qI_d + \frac{4kT}{R_L}} \quad (8.2)$$

$$\begin{aligned} P_{O/p}(dBm) = & P_{Laser} - P_{SMF-att} - P_{EC-IL} - P_{Si-att} \times N \times d_{MRR} \\ & - P_{MRM-IL} - (N-1)P_{MRM-OBL} - P_{splitter-IL} \times \log_2(M) \\ & - P_{MRR-W-IL} - (N-1)P_{MRR-W-OBL} - P_{penalty} - 10 \log_{10}(N) \end{aligned} \quad (8.3)$$

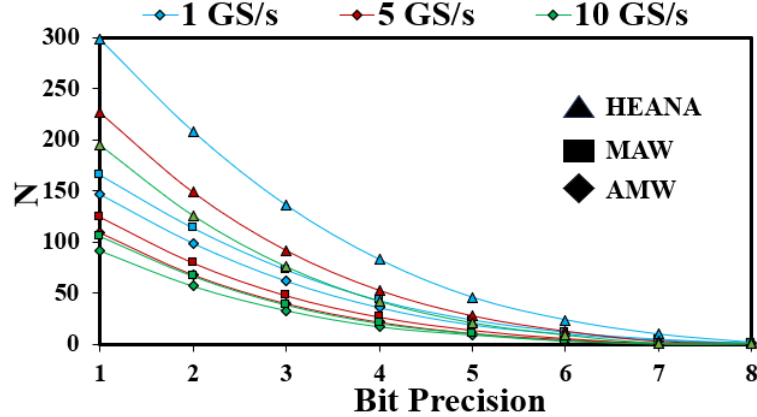


Figure 8.11: Supported DPU size N ($=M$) for bit precision= $\{1, 2, 3, 4, 5, 6, 7, 8\}$ bits at data rates (DRs)= $\{1, 5, 10\}$ GS/s, for AMW, MAW, and HEANA DPUs.

8.6 Evaluation

8.6.1 System Level Implementation of HEANA

Fig. 8.12 illustrates the system-level implementation of our HEANA accelerator. It consists of global memory that stores CNN parameters and a preprocessing and mapping unit. It has a mesh network of tiles. Each tile contains 4 DPUs interconnected

Table 8.2: Definition and values of various parameters used in Eq. 8.1, Eq. 8.2, and Eq. 8.3 (from [9, 152, 186]) for the scalability analysis.

Parameter	Definition	Value
P_{Laser}	Laser Power Intensity	10 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
P_{EC-IL}	Fiber to Chip Coupling Insertion Loss	1.44
$P_{MRR-W-IL}$	Silicon Waveguide Insertion Loss	0.3 dB/mm
$P_{splitter-IL}$	Splitter Insertion Loss	0.01 dB
P_{MRM-IL}	Optical Microring Modulator Insertion Loss	4 dB
P_{MRR-IL}	Optical Microring Resonator Insertion Loss	0.01 dB
$P_{MRM-OBL}$	Out of Band Loss	0.01 dB
$P_{Penalty}$	MAW Network Penalty	4.8 dB
	AMW Network Penalty	5.8 dB
	HEANA Network Penalty	1.8 dB

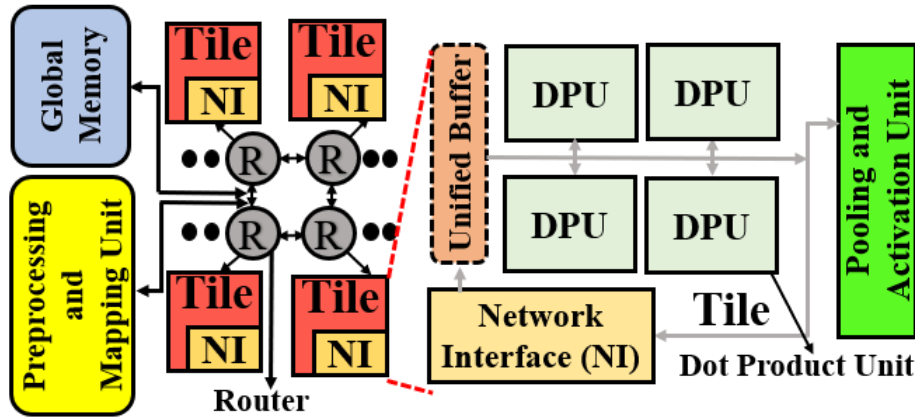


Figure 8.12: System-level overview of our HEANA accelerator.

(via H-tree) with a unified buffer, as well as pooling and activation units. Each DPU consists of multiple DPEs and each DPE is equipped with a dedicated input and output FIFO buffer [192] to store intermittent weights, inputs, and $psum$ results.

Table 8.3: DPU size (N) and DPU Count (#) at 4-bit precision across various DRs for different accelerators.

	Datarate					
	1 GS/s		5 GS/s		10 GS/s	
DPU	N	#	N	#	N	#
AMW	36	207	17	900	12	1950
MAW	43	280	21	1100	15	1610
HEANA	83	50	42	180	30	320

Table 8.4: Accelerator Peripherals and DPU Parameters [152]

	Power(mW)	Latency	Area(mm²)
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
DAC (ALL)¹ [181]	12.5	0.78ns	2.50E-3
DAC(HEANA)²[82]	26	0.78ns	6.00E-3
EO Tuning	80 μ W/FSR	20ns	-
TO Tuning	275 mW/FSR	4 μ s	-

8.6.2 Simulation Setup

For evaluation, we modeled our HEANA accelerator from Fig. 8.12 using our custom-developed, transaction-level, event-driven, python-based simulator. We simulated the inference of four CNNs (batch size=1 and batch size=256): GoogleNet [162], ResNet50 [65], MobileNet_V2 [138], and ShuffleNet_V2 [208]. We evaluate frames-per-second (FPS) and FPS/W (energy efficiency).

We compared our HEANA with the AMW [22] and MAW [109] accelerators. Our BPCA can be easily integrated into AMW and MAW accelerators. Therefore, we have considered two variants of AMW and MAW: (1) AMW and MAW (2) AMW integrated with BPCA (AMW_{BPCA}) and MAW integrated with BPCA (MAW_{BPCA}). Each accelerator variant is evaluated for the weight stationary (*WS*), input stationary (*IS*), and output stationary (*OS*) dataflows. All accelerators are operated for 4-bit integer precision across datarates 1GS/s, 5GS/s, and 10GS/s. From Fig. 8.11, for these parameters HEANA, AMW, and MAW achieve N reported in Table 8.3. For a fair comparison, we performed area proportionate analysis, wherein we scaled the DPU count for each photonic CNN accelerator so that the total area of DPUs in each accelerator matches with the area of HEANA ($N=83$) having 50 DPUs. Table 8.3 reports the scaled DPU count of AMW, MAW, and HEANA at various data rates. Table 8.4 gives the parameters used for our evaluation.

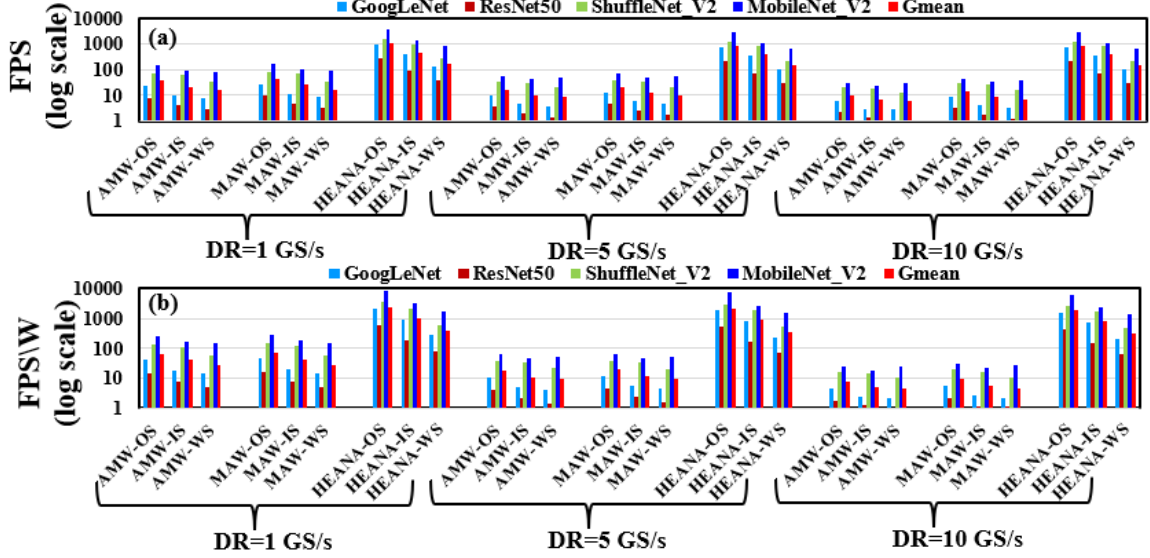


Figure 8.13: (a) Normalized FPS (log scale) (b) Normalized FPS/W for HEANA versus AMW and MAW accelerators with input batch size=1. Results of FPS and FPS/W are normalized with respect to AMW executing weight stationary dataflow (AMW-WS) for ResNet50 at 10 GS/s.

8.6.3 Evaluation Results

Fig. 8.13(a) shows normalized FPS results for various accelerators with batch size=1 at different data rates (DRs), normalized to AMW-WS for ResNet50 at 10 GS/s. Our accelerator HEANA outperforms MAW and AMW for the *IS*, *OS*, and *WS* dataflows across all data rates. At 1 GS/s, HEANA-OS achieves up to 30 \times and 25 \times better FPS than AMW and MAW, respectively, across all dataflows. As DR increases to 5 GS/s and 10 GS/s, our HEANA-OS further improves FPS, achieving up to 69 \times and 113 \times better FPS than AMW, respectively, across all dataflows. HEANA-OS achieves up to 55 \times and 83 \times better FPS than MAW at 5 GS/s and 10 GS/s, across all dataflows.

These significant improvements in throughput for HEANA are due to two reasons. First, our HEANA with spectrally hitless architecture supports a larger DPU size ($N=83$), i.e., the size of the spatial dot product operation N (refer Table 8.3) and the number of parallel dot product operations $M (=N)$, which increases the overall throughput with improved parallelism. Second, our BPCA eliminates the latency corresponding to *psum* reductions and corresponding buffer accesses for all dataflows as discussed in Sections 8.3.2 and 8.4.4.

Among dataflows, the *OS* dataflow achieves better throughput for HEANA compared to the *WS* and *IS* dataflows. For HEANA-OS, our BPCA activates the incoherent superposition [29] at BPD for accumulation (see Section 8.3.2), allowing TAOMs to operate at least 10 \times faster than the sample rate of the BPCA. Thus, HEANA-OS achieves FPS of up to 2.3 \times and 6.2 \times better than HEANA-IS and HEANA-WS, respectively, across all DRs.

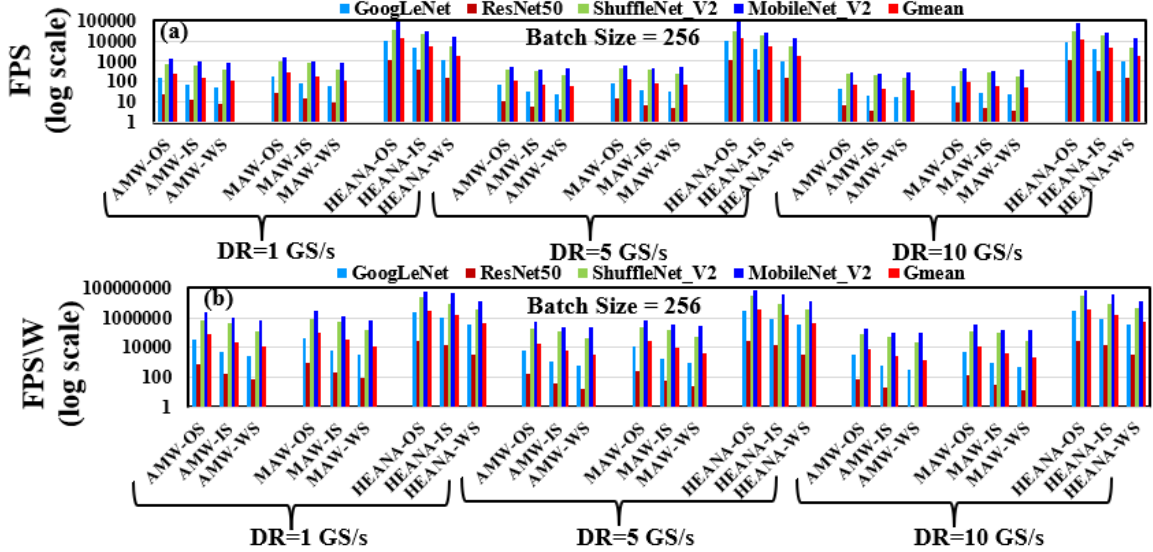


Figure 8.14: (a) Normalized FPS (log scale) (b) Normalized FPS/W for HEANA versus AMW and MAW accelerators with input batch size=256. FPS and FPS/W results are normalized with respect to AMW executing weight stationary dataflow (AMW-WS) for ResNet50 at 10 GS/s.

In the case of AMW and MAW architectures, the *OS* dataflow outperforms the *WS* and *IS* dataflow. The *OS* dataflow requires the fewest number of output buffer accesses compared to *IS* and *WS*. Additionally, the *OS* dataflow allows for the temporal accumulation of *psums* at a reduction network; therefore, consecutively arriving *psums* can be added together using a temporal accumulator [93] without having to store them temporarily in a buffer. Therefore, for AMW and MAW, the *OS* dataflow is better than the *IS* and *WS* dataflows. Furthermore, the *WS* dataflow achieves a lower FPS than the *IS* dataflow for AMW and MAW, as it incurs a higher buffer access latency due to more frequent buffer accesses.

Fig. 8.13(b) shows FPS/W (log scale) results for various accelerators with batch size=1 at different DRs, normalized to AMW-WS for ResNet50 at 10 GS/s. Our HEANA accelerator on gmean achieves better FPS / W across four CNNs and outperforms MAW and AMW for *IS*, *OS*, and *WS* dataflows across all data rates. At 1 GS/s, HEANA-OS on gmean achieves up to $36\times$ and $32\times$ better FPS/W than AMW and MAW, respectively, across all the dataflows. Similarly, at 5 GS/s and 10 GS/s, our HEANA-OS achieves up to $120\times$ and $244\times$ better FPS/W compared to AMW, across all dataflows. HEANA-OS achieves up to $104\times$ and $204\times$ better FPS/W compared to MAW at 5 GS/s and 10 GS/s, across all dataflows. HEANA-IS and HEANA-WS also achieve up to $137\times$ and $54\times$ better FPS/W compared to AMW and MAW across all the data rates. Our HEANA consumes less energy and static power compared to AMW and MAW. In HEANA, energy consumption related to *psum* buffer accesses is significantly reduced because of BPCA's in-situ spatio-temporal accumulations. The use of BPCA in HEANA also eliminates the energy consumption of the exter-

nal *psum* reduction network. As discussed in section 8.4, HEANA requires fewer ADC conversions, contributing to energy savings. Additionally, while AMW and MAW employ two MRRs to perform a single multiplication, HEANA employs a single MRR-based TAOM, thereby reducing static power consumption. These benefits collectively result in better FPS/W for HEANA-OS, HEANA-WS, and HEANA-IS. HEANA-OS achieves at least $2.1\times$ and $6\times$ better FPS/W than HEANA-WS and HEANA-IS across all the datarates. As DR increases, the FPS/W decreases across all the architectures due to increased energy consumption of ADCs and DACs [152].

Fig. 8.14(a) and Fig. 8.14(b) shows FPS (log scale) and FPS/W (log scale) results for various accelerators with batch size=256 at different DRs, normalized to AMW-WS for ResNet50 at 10 GS/s. Larger batch sizes yield significantly better benefits from HEANA, as HEANA-OS on gmean achieves up to $347\times$ better FPS than other architectures across all datarates. Similarly, as the batch size grows, the advantages of HEANA become more pronounced in FPS/W results. HEANA-OS on gmean achieves up to $952\times$ better FPS/W than other architectures across all the datarates. We found that the impact of dataflow choice is similar to that of batch size=1. Next we discuss the results of HEANA when MAW and AMW are integrated with BPCA.

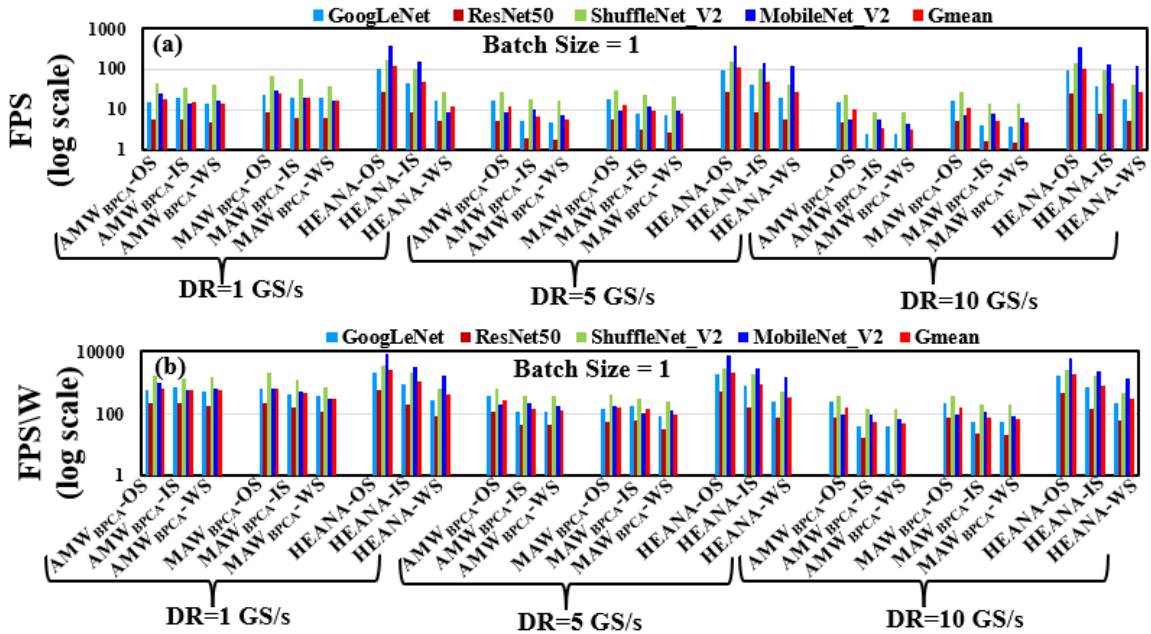


Figure 8.15: (a) Normalized FPS (log scale), (b) Normalized FPS/W for HEANA versus BPCA-integrated versions of AMW and MAW accelerators with input batch size=1. Results of FPS and FPS/W are normalized with respect to AMW executing input stationary dataflow (AMW_{BPCA-WS}) for ResNet50 at 10 GS/s.

Fig. 8.15 (a) shows FPS (log scale) results for HEANA versus BPCA integrated AMW_{BPCA} and MAW_{BPCA} accelerators with batch size=1 at different DRs, normal-

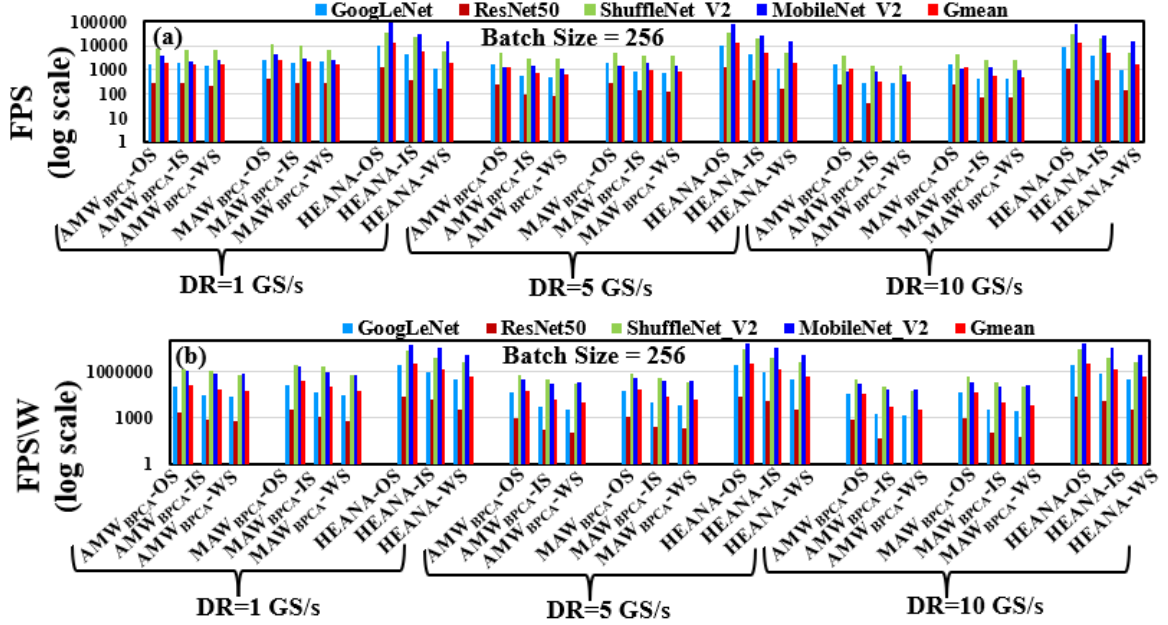


Figure 8.16: (a) Normalized FPS (log scale), (b) Normalized FPS/W for HEANA versus BPCA-integrated versions of AMW and MAW accelerators with input batch size=256. Results of FPS and FPS/W are normalized with respect to AMW executing input stationary dataflow ($AMW_{BPCA-WS}$) for ResNet50 at 10 GS/s.

ized to AMW_{WS} for ResNet50 at 10 GS/s. Our accelerator HEANA on gmean across four CNNs, outperforms AMW_{BPCA} and MAW_{BPCA} for *IS*, *OS*, and *WS* dataflows across all datarates. At 1 GS/s, HEANA-OS on gmean achieves up to $6.3\times$ and $4.6\times$ better FPS than AMW_{BPCA} and MAW_{BPCA} , respectively, across all the dataflows. At higher datarates such as 5 GS/s and 10 GS/s, our HEANA-OS achieves up to $8\times$ and $9\times$ better FPS than AMW_{BPCA} and MAW_{BPCA} , across all dataflows. We can observe that the throughput gap between HEANA and BPCA-integrated AMW and MAW architectures decreases, with the integration of BPCA, AMW_{BPCA} and MAW_{BPCA} architectures leverage in-situ temporal accumulations at BPCA reducing the latency corresponding to the reduction of *psums*. Therefore, AMW_{BPCA} and MAW_{BPCA} achieve better FPS than baseline AMW and MAW. With the integration of BPCA, the throughput order of dataflows remains unchanged in AMW_{BPCA} and MAW_{BPCA} . Although *IS* and *WS* dataflow exploit BPCA in-situ accumulation, they still need frequent switching of capacitors at BPCA to reduce *psums* corresponding to various output pixels. In contrast, *OS* dataflow performs temporal accumulation without requiring capacitor switching at BPCA (refer Section 8.4), thus giving better throughput than *IS* and *WS* dataflow.

Fig. 8.15(b) shows FPS/W (log scale) results for HEANA versus BPCA integrated AMW_{BPCA} and MAW_{BPCA} accelerators with batch size=1 at different DRs, normalized to AMW_{WS} for ResNet50 at 10 GS/s. Our accelerator HEANA on gmean achieves better FPS/W across four CNNs, outperforms AMW_{BPCA} and MAW_{BPCA}

for IS, OS, and WS dataflows across all datarates. At 1 GS/s, HEANA-OS on gmean achieves up to $5.4\times$ and $3.6\times$ better FPS/W than AMW_{BPCA} and MAW_{BPCA} , respectively, across all the dataflows. At higher datarates such as 5 GS/s and 10 GS/s, our HEANA-OS achieves up to $35\times$ and $26\times$ better FPS than AMW_{BPCA} and MAW_{BPCA} , across all dataflows. As mentioned before, HEANA’s TAOMs reduce the static power consumption of MRRs by using a single MRR design which significantly reduces the power consumption.

Fig. 8.16(a) and Fig. 8.16(b) show FPS (log scale) and FPS/W (log scale) results for HEANA versus BPCA-integrated versions of AMW_{BPCA} and MAW_{BPCA} accelerators with batch size=256 at different DRs, normalized to AMW-WS for ResNet50 at 10 GS/s. HEANA-OS on gmean achieves up to $23\times$ better FPS than other architectures across all datarates. Similarly, HEANA also achieves higher FPS/W for batch size=256. HEANA-OS on gmean achieves up to $92\times$ better FPS/W than other architectures across all the datarates. We found that the impact of dataflow choice is similar to that of batch size=1.

The area efficiency values (FPS/W/mm²) for each accelerator across various CNNs are similar to the energy efficiency (FPS/W) values for area proportional analysis (when the area of all the accelerators is matched to the area of HEANA). Therefore, we have not reported area efficiency results. Overall, HEANA significantly improves throughput (FPS) and energy efficiency (FPS/W) across various data rates and dataflows compared to the tested analog optical accelerators.

8.6.4 Inference Accuracy Results

As discussed in Section 8.3.2, HEANA provides several advantages compared to prior analog accelerators, but TAOMs of HEANA incur some inaccuracies in dot product computation (see Section 8.3.2) due to the possible calibration errors and analog noise. To evaluate the impact of these inaccuracies on CNN inference, we evaluated the inference accuracy of CNN models on HEANA. We integrated our custom simulator with ML-framework PyTorch [126] and incorporated a mean absolute error model of TAOM-based multiplication operation (see Section 8.3.2) in the simulator. Then, using this simulator, we performed inference on ImageNet validation dataset [171] (50k images and 1k classes). Table 8.5 reports the Top 1 and Top 5 inference accuracy across various 8-bit quantized CNN models on the analog architectures and HEANA. Our HEANA accelerator results in Top 1 and Top 5 errors of 0.1% and 0.1%, respectively, on average across CNN models. Our HEANA accelerator’s excellent gains in FPS and FPS/W make this minor drop in CNN inference accuracy tolerable.

8.7 Discussion

HEANA presents several innovations at the circuit level and architecture level to mitigate the shortcomings of prior optical CNN accelerators. At the circuit level, HEANA employs a novel time-amplitude analog optical modulator (TAOM) that generates the product of one input value and weight value as a pulse-width-amplitude-modulated

Table 8.5: Top-1 and Top-5 inference accuracy comparison of HEANA versus MAW for 8-bit quantized CNNs {GoogleNet (GNet), ResNet50 (RNet50), MobileNet_V2 (MNet_V2), ShuffleNet_V2 (SNet_V2)} and ImageNet dataset [45].

HEANA ACCURACY DROP (%)	GNet [162]	RNet [65]	MNet_V2 [138]	SNet_V2 [208]	Gmean
TOP-1	0	0	0.1	0.1	0.1
TOP-5	0.1	0.1	0.1	0.1	0.1

(PWAM) symbol whose optical energy is proportional to the product result. TAOM achieves this by employing a single microring as opposed to the dual microrings used by prior works[22, 157, 109]. By using a single microring for each multiplication, HEANA reduces the needed feedback control units to half, which decreases the static power consumption and improves the energy efficiency. Use of single microring per multiplication also reduces the overall optical losses and power penalties, furthering the energy efficiency advantage. In addition, HEANA also employs a balanced photo-charge accumulator (BPCA) which leverages the capabilities of a balanced photodetector to perform in-situ spatio-temporal accumulations. As discussed in Section 8.4, BPCA improves power consumption by reducing the analog-to-digital conversions and buffer accesses, and by completely eliminating the use of a reduction network for accumulations. With all these cumulative advantages HEANA gains at least $32\times$ better energy efficiency than prior optical accelerators. At the architectural level, HEANA presents a spectrally hitless architecture that mitigates inter-modulation crosstalk which reduces the power penalty, increasing the available power budget. The saved power budget allows HEANA to improve the DPU size from 44×44 to 83×83 at 4-bit precision, this improves spatial parallelism and decreases the number of *psum* reductions required. This in turn significantly decreases the latency corresponding to the analog-to-digital conversions and *psum* reductions to complete the computations by reducing the required analog-to-digital conversions and *psum* reductions. These benefits allow HEANA to achieve at least $23\times$ better throughput than prior optical accelerators. Thus, due to these cross-layer innovations, HEANA achieves superior performance compared to prior optical CNN accelerators.

8.8 Summary

In this Chapter, we presented HEANA, a novel hybrid time-amplitude analog optical GEMM accelerator. The dot product element of our HEANA employs a spectrally hitless array of novel hybrid time-amplitude analog optical modulators (TAOM) that perform multiplication operations, combined with a balanced photo-charge accumulator (BPCA) that performs spatio-temporal accumulation of the individual multiplication results from TAOMs. Each TAOM utilizes a single active microring resonator (MRR) to perform multiplication operations, allowing for the electro-optic tuning of both input and weight values. This enables the flexibility of executing various

CNN dataflows namely input stationary, output stationary, and weight stationary dataflows. Moreover, The spectrally hitless arrangement of TAOMs in HEANA eliminates spectral-interference and various crosstalk effects encountered by current MRR-enabled analog photonic accelerators. In addition, the single MRR implementation of TAOMs substantially reduces the area consumption and the insertion losses in HEANA, thereby increasing its energy efficiency. Also, the BPCA circuit in HEANA enables in-situ accumulation of a large number of partial sums, thereby reducing the overall latency and energy consumption of CNN processing.

We performed detailed modeling and characterization of our TAOM unit using photonics foundry-validated tools from ANSYS/Lumerical. Here, we performed time-domain simulations from which we evaluated the performance of our TAOM in terms of accuracy and precision. We have also performed a scalability analysis of our HEANA's dot product units, to determine their achievable maximum size and supported bit-precision. Finally, we evaluated HEANA for input stationary, output stationary, and weight stationary dataflows at the system-level, and compared its performance with two well-known photonic CNN accelerators from prior works. Our system-level evaluation results for four CNN models show that HEANA provides improvements of up to $25\times$ and $32\times$ in throughput, and energy efficiency, respectively, compared to two prior optical analog accelerators AMM and MAM, with Top-1 accuracy drop of only up to 0.1%.

Acknowledgments

We would like to acknowledge the National Science Foundation (NSF) as this research was supported by NSF under grant CNS-2139167.

Chapter 9 Stochastic Computing based Optical Accelerator with Functional Reconfigurability for efficient inference of Heterogeneous Quantized CNNs

9.1 Introduction

Convolutional Neural Networks (CNNs) have transformed various artificial intelligence tasks, including image recognition, language translation, and autonomous driving [96, 47], due to their exceptional inference accuracy. As CNNs are applied to tackle progressively intricate tasks, their demand for computational power and memory has correspondingly escalated. The substantial computational and storage demands of CNNs hinder their practical application. Thus, to enhance the speed and efficiency of CNN inference, model compression methods such as quantization are widely utilized [61, 214, 56]. Quantization techniques enable the creation of compact CNNs compared to their floating-point equivalents by representing the weights/inputs of CNNs with lower precision. These quantized techniques can be categorized into three main types: homogeneous quantization, heterogeneous quantization, and binary quantization. Homogeneous quantization assigns the same fixed precision to most or all layers of a CNN, simplifying implementation but potentially sacrificing accuracy. Heterogeneous quantization adjusts the precision of each layer based on its impact on accuracy, aiming to balance model size and inference accuracy. Binary quantization reduces both input and weight precision to 1-bit, significantly reducing memory footprint and computational complexity but potentially compromising accuracy, particularly for complex tasks. Overall, quantization offers a dual benefit by reducing both memory usage and energy requirements for CNN inference.

In addition to these techniques, the ever-increasing complexity and inference time of CNNs has pushed for highly customized CNN hardware accelerators [20]. Among the CNN hardware accelerators from the literature, silicon-photonics accelerators have shown great promise to provide unparalleled parallelism, ultra-low latency, and high energy efficiency [109, 58, 186, 22, 34, 157]. Typically, a silicon-photonics CNN accelerator consists of multiple Processing Units (PUs) that perform multiple dot product operations in parallel. Several optical CNN accelerators have been proposed in prior works based on various silicon-photonics devices, such as Mach Zehnder Interferometer (MZI) (e.g., [206], [44]) and Microring Resonator (MRR) (e.g., [22], [157]).

Among these optical CNN accelerators from prior work, the MRR-enabled analog optical accelerators (e.g., [109, 22, 58, 145, 157, 166]) have shown disruptive performance and energy efficiencies, due to the MRRs' compact footprint, low dynamic power consumption, and compatibility with cascaded dense wavelength division multiplexing (DWDM). However, these accelerators face several challenges that hinder their scalability, throughput, and energy efficiency. These prior accelerators employ a combination of microring modulators (MRM) input array and MRR weight bank to perform dot product operation between input and weight values. The intermodulation crosstalk in MRM input array and inter-spectral, electrical, and thermal

crosstalk effects in MRR weight banks reduce the available optical power budget in PUs. The reduction in the optical power budget significantly reduces the achievable PU size and supported bit-precision [9, 152]. In addition, in these accelerators, cascaded dense wavelength division multiplexing also limits the scalability due to limited free spectral range [9]. Moreover, the proposed accelerators were tailored for either homogeneous quantized CNNs or binary quantized CNNs, making them unsuitable for executing the alternative type of CNNs due to differing computational requirements. Binarized CNNs a.k.a Binary neural networks (BNNs) necessitate XNOR operations, while homogeneous quantized CNNs rely on dot product operations, leading to a mismatch in functionality. Furthermore, the development of accelerators for processing heterogeneous quantized CNNs has largely been overlooked in the literature [156]. Accelerators designed for homogeneous quantized CNNs are inefficient at processing heterogeneous quantized CNNs because they do not effectively utilize the advantages of low precision layers, instead they process them at high precision. Additionally, none of the MRR-based CNN accelerators proposed to date support the execution of all types of quantized CNN models: homogeneously quantized, heterogeneously quantized, and binary quantized. Besides, prior accelerators have not exploited the application of homodyne incoherent superposition at photodetectors for accumulation.

To address these shortcomings, this Chapter presents a novel Stochastic Computing based Optical Accelerator with Functional Reconfigurability (SCOAR). SCOAR employs a novel design of reconfigurable MRR-based logic gates (RLGs) in a spectrally hitless arrangement to eliminate spectral-interference and various crosstalk effects. Our RLG uses a single MRR to perform stochastic multiplication or XNOR operation depending on the quantized CNN requirements. SCOAR also employs precision adaptive input peripherals to dynamically change the supported precision to efficiently process heterogeneously quantized CNNs. Moreover, SCOAR employs a balanced photocharge accumulator, which inherently supports the accumulation of a very high number of *psums*, thereby eliminating the need of using external *psum* reduction networks, to consequently reduce the overall latency and energy consumption of quantized CNN processing.

Our key contributions in this Chapter are summarized below:

- We present our invented, novel, functionally flexible CNN accelerator called SCOAR, which employs an array of MRR based reconfigurable logic gates (RLG) in spectrally hitless PU architecture and highly scalable in-situ spatio-temporal accumulators called Balanced Photo-Charge Accumulators (BPCAs);
- We present detailed modeling and characterization of our invented RLG using foundry-validated, commercial-grade, photonic-electronic design automation tools (Section 9.4.3);
- We employ our designed RLGs and BPCAs to forge a highly scalable and functionally flexible CNN accelerator named SCOAR, which employs RLG and BPCA-based scalable PUs (Section 9.4);

- We perform a comprehensive scalability analysis for our SCOAR PUs, to determine their achievable maximum size \mathbf{N} , operating speed, and error susceptibility (Section 9.5);
- We implement and evaluate SCOAR at the system-level using our in-house simulator, and compare its performance and inference accuracy for processing all types of quantized CNNs with four MRR based CNN accelerators from prior works (Section 8.6.3).

9.2 Preliminaries

9.2.1 Quantization of Convolutional Neural Networks

Convolutional Neural Networks (CNNs) have become the standard deep learning architecture for achieving remarkable accuracy in computer vision tasks. However, the continual growth in model sizes poses significant challenges for hardware resources, including storage and computational requirements, during both training and inference stages [20]. In recent years, research efforts in both software and hardware have focused on the quantization of CNNs [106] to perform low-precision inference. Generally, 32-bit (full-precision) floating point numbers are used to represent weights (W) and inputs (I) of CNNs. Quantization converts the full-precision weights and inputs to fixed-point numbers with lower bit-precision, such as 8, 4, and 1-bit [74]. The conversion reduces memory usage by storing values in low bit-precision, and it also decreases FLOPs by replacing high-cost floating-point operations with low-cost fixed-point operations. **Homogeneous quantization** methods use the fixed precision for all (or most of) the layers of CNNs. However, such uniform bit-precision assignment can be suboptimal since quantizing different layers can have a different impact on the accuracy and efficiency of the overall network. For example, among all the layers of a CNN, the first and last layers cause more accuracy drop in quantization [205] whereas other layers are less sensitive to quantization. **Heterogeneous quantization** methods, consider this variation of sensitivity towards quantization in layers and quantize different layers to lower bit widths based on their impact on accuracy. Such quantization methods [195] result in mixed-precision heterogeneous quantized CNNs with variations in inter-layer bit-precision requirements. These heterogeneous quantized CNNs [195] require lower memory and computation requirements than fixed precision quantized CNNs. Furthermore, **binary quantization** represents an extreme form of quantization wherein both input and weight values are quantized to 1-bit precision [132]. This approach significantly reduces memory usage and simplifies computation by relying on basic logical XNOR operations. Networks employing this technique are often termed Binary Neural Networks (BNNs). However, it's important to note that not all Convolutional Neural Network (CNN) models can be effectively binarized or heterogeneously quantized without experiencing a notable decrease in accuracy.

9.2.2 Processing Quantized CNNs on Hardware

In CNNs, the major computational requirement arises from convolutional layers. These layers involve convolution operations that can be converted to General Matrix-Matrix Multiplication (GEMM) operations using the Toeplitz matrix or the im2col transformation [161, 3]. As shown in Fig. 8.1, the input feature map (Fmap) belonging to a convolution layer is unfolded into the matrix \mathbf{I} . The weight filters of the convolution layer are flattened and stacked to form the weight matrix (\mathbf{W}). The GEMM operation between \mathbf{I} and \mathbf{W} gives the resultant output matrix (\mathbf{O}). On conventional CPUs/GPUs, GEMM operations are mapped and executed using basic linear algebra subprograms (BLAS) or Cuda BLAS (cuBLAS) [27, 50]. However, conventional CPUs/GPUs cannot efficiently meet the exponentially growing computational demand of modern CNNs. To meet this demand, both industry and academia have proposed various dedicated GEMM accelerators [81, 157, 22, 94], tailored to process CNNs with better performance and energy efficiency.

The dedicated GEMM accelerators decompose GEMM operations into dot products between the rows of the input matrix and the columns of the weight matrix. These dot products, of size (S), are then mapped onto the accelerators based on the supported size (N) of the accelerator. For layers where $S > N$, the accelerators decompose the dot product into multiple dot products each of size N . These dot products generate partial sums (*psums*), which are later added using a reduction network to obtain the final dot product result of size S . In addition to the size requirement, the accelerators also need to consider the precision requirement of CNNs. To perform inference on homogeneously quantized CNNs, the accelerators can process these dot products without preprocessing when the required precision of the quantized CNN is supported by the accelerator. Otherwise, the accelerator employs different preprocessing techniques such as bit slicing to achieve the required precision. However, these preprocessing steps incur additional overhead in terms of latency and energy consumption. In the case of heterogeneously quantized CNNs, the precision requirement can vary from layer to layer of the CNN. To support such variations, accelerators with flexible precision can reconfigure to meet the requirement, while accelerators without precision reconfigurability will process the layers at their maximum supported precision, thereby diminishing the benefits of heterogeneously quantized CNNs. Furthermore, for binary quantized BNNs, the computation requirement shifts from dot products to XNOR-bit count operations [132].

9.2.3 Related Work on Optical CNN 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., [109, 157, 22, 44, 186]). These accelerators employ PIC-based Processing Units (PUs) to perform multiple parallel dot product operations. Some accelerators implement digital PUs (e.g., [145, 94, 186]), whereas some others employ analog PUs (e.g., [109, 157, 22, 152]). Different PU implementations employ MRRs (e.g., [109, 22, 157, 111, 186]) or MZIs (e.g., [44, 16, 206]) or both (e.g., [145], [94]). The

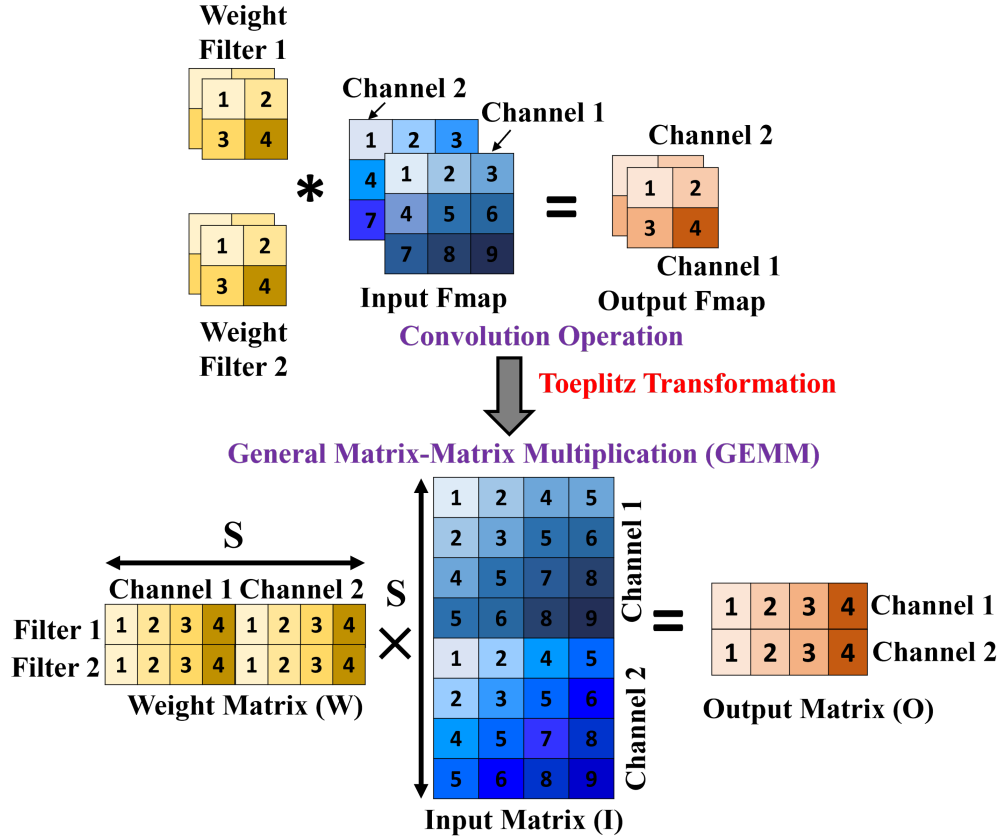


Figure 9.1: Convolution operation at a convolution layer with two weight filters and one input feature map (Fmap) having two channels is transformed into a GEMM operation between input matrix I and weight matrix W .

analog PUs can be further classified as incoherent (e.g., [109, 157, 22]) or coherent (e.g., [63, 211, 199]). To set and update the values of the individual input and weights used for dot product operations, the incoherent PUs utilize the analog optical signal power, whereas the coherent PUs utilize the electrical field amplitude and phase. The coherent PUs achieve low inference latency, but they suffer from control complexity, high area overhead, low scalability, low flexibility, high encoding noise, and phase error accumulation issues [119]. In contrast, the MRRs-enabled incoherent PUs achieve better scalability and lower footprint, because they use PICs that are based on compact MRRs [22], unlike the coherent PUs that use PICs based on bulky MZIs. Various state-of-the-art PIC-based optical CNN accelerators are well discussed in a survey paper [43]. Because of the inherent advantages of MRR-enabled incoherent PUs, there is an impetus to design more energy-efficient and scalable CNN accelerators employing MRR-enabled incoherent PUs.

The prior MRR-enabled incoherent PUs proposed in [22, 109] exhibit a significant trade-off between achievable precision (b) and PU size (N). It has been demonstrated in [152] that the feasible size of analog DPUs is constrained to $N < 44$ for bit precision $B > 4$. Typically, state-of-the-art homogeneously quantized CNNs adopt 8-bit integer

precision for both weights (W) and inputs (I) to maintain accuracy. However, analog PUs cannot support $b=8$ for $N > 1$ [152], hence they are designed with $b=4$ and N bit-slicing to facilitate dot product operations for models with W and I values exceeding b . Previous works [152, 22, 109] employ bit-slicing of weights and inputs, along with bit shifting and accumulation, to achieve high-precision multiplication with analog PUs supporting low precision.

For the processing of homogeneously quantized CNNs, SCONNA [186] addressed the trade-off challenge between b and N in analog incoherent MRR-based PUs by integrating stochastic computing with photonic computing. Additionally, several MRR-enabled incoherent PUs specialized in processing BNNs were proposed [215], [188], and [160]. Furthermore, to efficiently handle heterogeneous quantized CNNs, an MRR-enabled incoherent PU with a time division multiplexing scheme was proposed in HQNNA [156]. However, none of the MRR-based accelerators from previous works effectively process all homogeneously, heterogeneous, and binary quantized CNNs.

9.2.4 Stochastic Computing

Stochastic Computing (SC) is an unconventional form of computing that represents and processes data in the form of probabilistic values called stochastic numbers (SNs) [54, 10, 15]. In SC’s unipolar format, an SN W is a bit-stream of N bits that represents a real-valued variable $v \in [0, 1]$ by encoding v through the ratio N_1/N , where N_1 is the number of 1’s in W . SC offers several advantages over conventional binary computing such as high error tolerance, low power consumption, small circuit area, and low-cost arithmetic operations consisting of standard digital logic components [15]. For example, multiplication can be performed by a stochastic circuit consisting of a single AND gate.



Figure 9.2: Multiplication between unipolar stochastic numbers I and W .

9.3 Motivation

Research on MRR-based incoherent processing units has primarily focused on the inference of homogeneous quantized CNN models [157, 22, 186, 145]. Optical accelerators designed for executing homogeneous quantized models [22, 157, 186] fail to leverage the advantages of heterogeneous quantized models due to their fixed precision of PUs. With fixed precision support, these accelerators end up using resources necessary for processing the highest supported precision, thereby diminishing the benefits in latency and power consumption offered by heterogeneous quantized models. Nevertheless, few works have proposed specific accelerators for efficiently executing BNN

models [188, 160, 215]. However, these accelerators either cannot execute models with precision higher than 1-bit or require significant preprocessing involving bit slicing, resulting in low throughput and energy efficiency for the execution of heterogeneous quantized CNNs.

Furthermore, to our knowledge, only one incoherent MRR-based optical accelerator has been proposed to execute heterogeneous quantized CNNs [156]. Although [156] can support the execution of heterogeneous quantized models with a time division multiplexing scheme, the scheme requires frequent use of ADCs, shifters, and adders, which contribute significantly to energy consumption and latency. Additionally, all these accelerators, except [188], have two common shortcomings. Firstly, they consist of analog PUs, which inherently limit the achievable size of PUs due to the tradeoff between achievable size and supported precision [9, 152], thus limiting parallelism. Secondly, due to the lack of in-situ temporal accumulators, they convert each analog *psum* into digital and perform accumulation in the digital domain. This kind of *psum* accumulation increases latency and energy consumption due to the involved ADC conversions and digital *psum* accumulations with an electronic reduction network.

To address these challenges, prior work [186] designed a stochastic computing integrated digital optical accelerator with photo charge accumulators. With the integration of stochastic computing, SCONNA improved the achievable size limit of optical MRR-based accelerators and performed in-situ temporal accumulations. However, SCONNA is also designed for the inference of homogeneous quantized CNNs at 8-bit fixed precision and hence ends up using full precision resources while processing heterogeneous quantized CNNs. Moreover, it cannot process BNNs due to its functionally fixed processing elements (PEs) that cannot perform the XNOR operation required by BNNs. Additionally, SCONNA and all prior works fail to address two shortcomings of optical accelerators. Firstly, all these accelerators use dense wavelength division multiplexing with cascaded MRRs in their PUs, which incurs a high power penalty due to inter-modulation crosstalk in MRM input arrays [152, 88] and inter-spectral, electrical, and thermal crosstalk effects in MRR weight banks [167], ultimately reducing the available optical power budget in PUs. A reduced power budget limits scalability and thus the parallelism obtained by MRR-based optical accelerators. Besides, the cascaded DWDM based PE also enforces the FSR limited N on scalability. Secondly, prior works propose to employ on-chip laser diodes as the source of optical wavelengths used for DWDM; however, integrating a total of 176 LDs as proposed in SCONNA [186] or 50 LDs as proposed in HQNNA [156] onto a photonic integrated circuit is highly challenging and practically infeasible.

To address all these challenges in processing all types of quantized CNNs, we propose an MRR-based optical reconfigurable logic gate (RLG) and employ multiple RLGs to develop a novel Stochastic Computing-based optical accelerator with functional reconfigurability (SCOAR). The following section discusses our SCOAR architecture.

9.4 Our Proposed Architecture

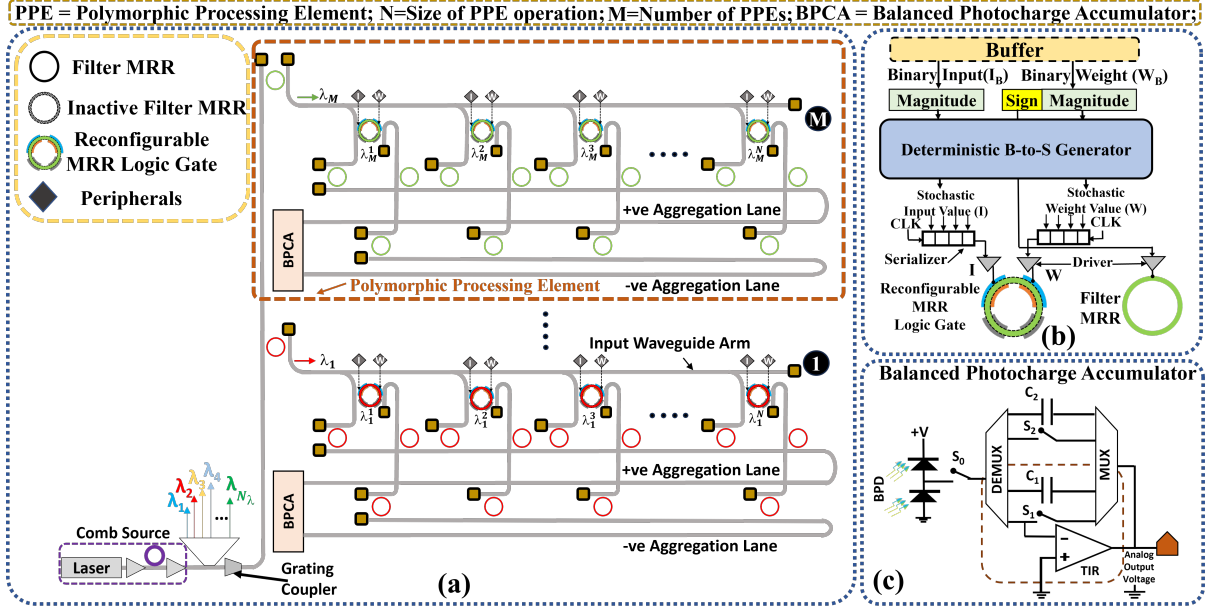


Figure 9.3: Schematics of (a) Our SCOAR-Stochastic Computing based Optical Accelerator with functional reconfigurability (b) Pheriperals of reconfigurable logic gate (c) Balanced Photocharge Accumulator.

9.4.1 Overview of our SCOAR Processing Unit

Fig. 9.3(a) illustrates the processing unit organization of our SCOAR architecture. A SCOAR processing unit consists of multiple processing elements that are polymorphic and reconfigurable in nature. Each processing unit carries out operations in parallel. For that, a single-on-chip comb source laser with a total N comb wavelengths is used, with each wavelength of power $P_{\lambda_i}^{in}$. These comb wavelengths get filtered into M input waveguide arms (IWAs). Every IWA receives single wavelength optical power λ_i with the help of filter MRRs and each IWA further splits the optical power of λ_i into N $\lambda_i^1 - \lambda_i^N$ signals and feeds N spectrally hitless reconfigurable logic gates. There are a total of M IWAs and M polymorphic processing elements (PPEs) in the SCOAR PU(Fig. 9.3(a)).

Each PPE consists of four components: (i) an array of N reconfigurable MRR logic gates (RLGs) arranged in a spectrally hitless manner; (ii) adaptive precision enabling stochastic peripherals; (iii) a bank of filter MRRs; and (iv) a Balanced Photo-Charge Accumulator (BPCA). According to the processing requirements of the quantized CNN, each PPE can operate in two modes: dot product operation mode for homogeneous and heterogeneous quantized CNNs, and XNOR-bit count operation mode for BNNs. Depending on the operation mode, each RLG performs

stochastic multiplication or XNOR operation between an input bit-stream I and weight bit-stream W .

Each RLG receives its bit-streams I and W from its corresponding peripherals at a supported bitrate (BR). Bit-stream W provides a weight value along with a sign bit. Bit-stream I provides the RELU-activated output value from the previous CNN layer, without a sign bit as RELU has a non-negative output. The detailed design of $RLGs$ and their peripherals is explained in Section 9.4.3. Each RLG performs a bit-wise logical AND (or logical XNOR) operation between the I and W bit-streams to produce a resultant optical bit-stream that represents the stochastic multiplication (XNOR operation) between the I and W bit-streams. The optical bit-streams from the RLGs, with each bit-stream carrying a stochastic multiplication (XNOR operation) result, reach the filter MRRs on drop port(through port) of RLGs. The application of filter MRRs is determined based on an operation mode, for dot product operation mode, the resultant bit-stream is generated at drop port and the filter MRRs on the through port of RLGs are turned off. Each drop port in RLG employs two filter MRRs, the first MRR receives the sign bit from the peripheral W of its corresponding RLG (Fig. 9.3(a)). The sign bit operates the filter to steer the incoming optical bit-stream λ_i^j to the +ve aggregation lane (if the sign bit is '0') or to the second MRR filter that guides the results to -ve aggregation lane (if the sign bit is '1'). Thus, the +ve aggregation lane and -ve aggregation lane of a PPE guide the optical bit-streams, carrying the stochastic multiplication results, to BPCAs. For XNOR-bit count operation mode, the resultant bit-stream is generated at the throughport of RLGs. Therefore, in XNOR-bitcount operation mode the filter MRRs on the drop port are turned off. Each throughport port in RLG employs a single filter MRR, this MRR always filters the resultant XNOR operation bit stream to the +ve aggregation lane. The +ve aggregation lane of a PPE guides the optical bits, carrying the XNOR operation results, to BPCAs. A BPCA is a circuit that collects all the optical bit-streams (i.e., stochastic multiplication results or XNOR operation results) from its corresponding +ve aggregation lane and -ve aggregation lane to generate the accumulation result or bit count value in the binary format (details about BPCA in Section 9.4.5).

9.4.2 Operational Modes of SCOAR Processing Element

Fig 9.5 shows the two operational modes of our SCOAR polymorphic processing element (PPE). Our SCOAR PPE can be operated in two modes (i) dot product operation and (ii) XNOR-bitcount operation. As discussed in Section 9.2.2, the homogenous quantized, heterogenous quantized and binarized CNN models have different computation requirements.

Dot Product Mode: By default, SCOAR PU is configured to operate in the dot product operation mode as shown in Fig. 9.5(a). The dot product operation mode is employed to support the processing of mixed-precision CNN models with varying precision across the different layers of the model. At each layer, the GEMM operation between I and W is mapped onto SCOAR PU as dot product operations, and each PPE in the PU performs a dot product operation of size N . In a PU, M

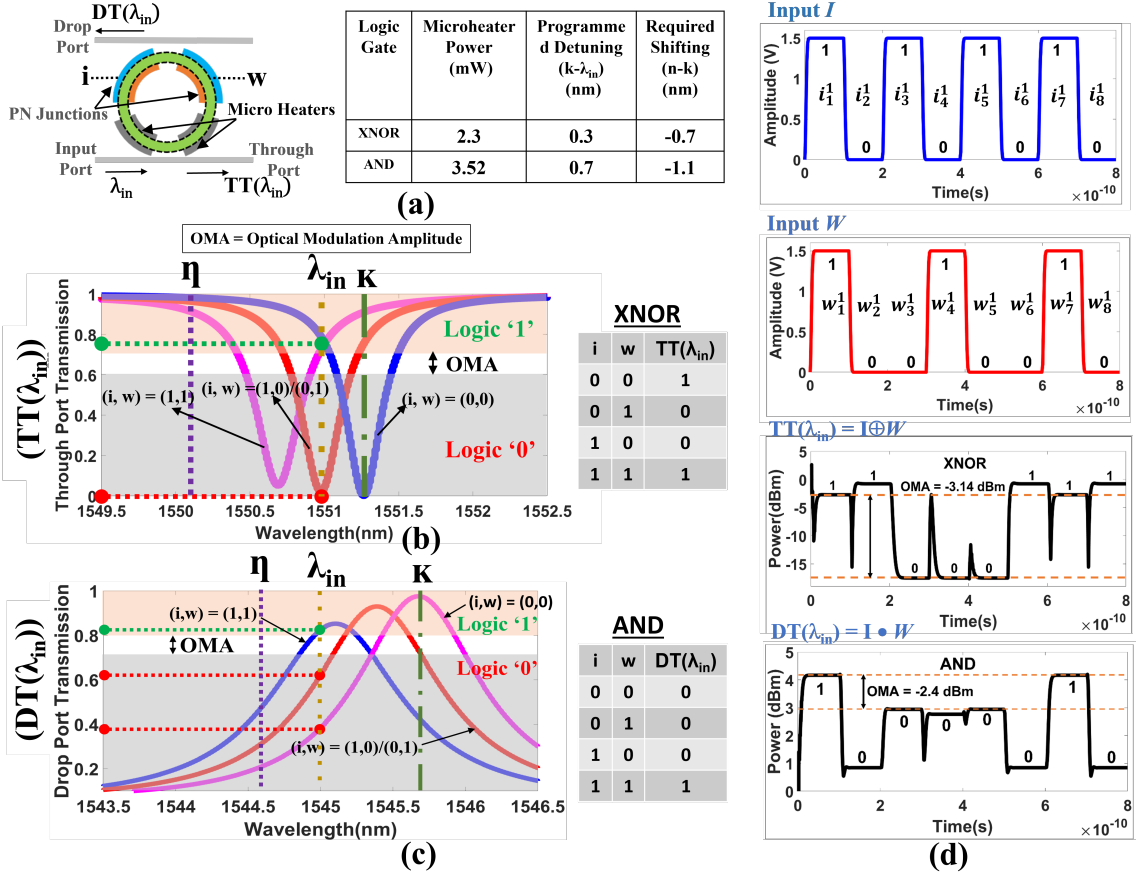


Figure 9.4: (a) Schematic of our Optical reconfigurable logical Gate (RLG) (b) operation of RLG as XNOR gate (c) operation of RLG as AND gate (d) results of RLGs transient analysis

PPEs work in parallel performing M parallel dot product operations. At each PPE, precision adaptive input peripherals convert the b -bit binary values I_b and W_b into stochastic bitstreams I and W of length $L=2^b$. The stochastic bitstreams are fed to the N hitlessly arranged RLGs, in dot product mode, RLGs are configured as logical AND gates to realize stochastic multiplication as discussed in Section 9.2.4. The RLGs generate the output of stochastic multiplication at their drop ports. Each drop port waveguide is equipped with two MRR filters to transfer the result onto +ve and -ve aggregation lanes depending on the sign of the W_b . In addition, the MRR filters on the throughport of RLGs are turned off. The aggregation lanes guide the resultant bits of stochastic multiplication to BPCA for accumulation.

XNOR-bitcount Mode: Fig. 9.5(b) shows the operation of SCOAR PPE in XNOR-bitcount mode. As discussed in Section 9.2.2, BNN processing requires XNOR operations followed by bitcount. For XNOR-bitcount mode, the input peripherals of PPE fed the 1-bit I and W values to RLGs, here RLGs are programmed to operate as logical XNOR gates. The RLGs generate the output of XNOR operation at the

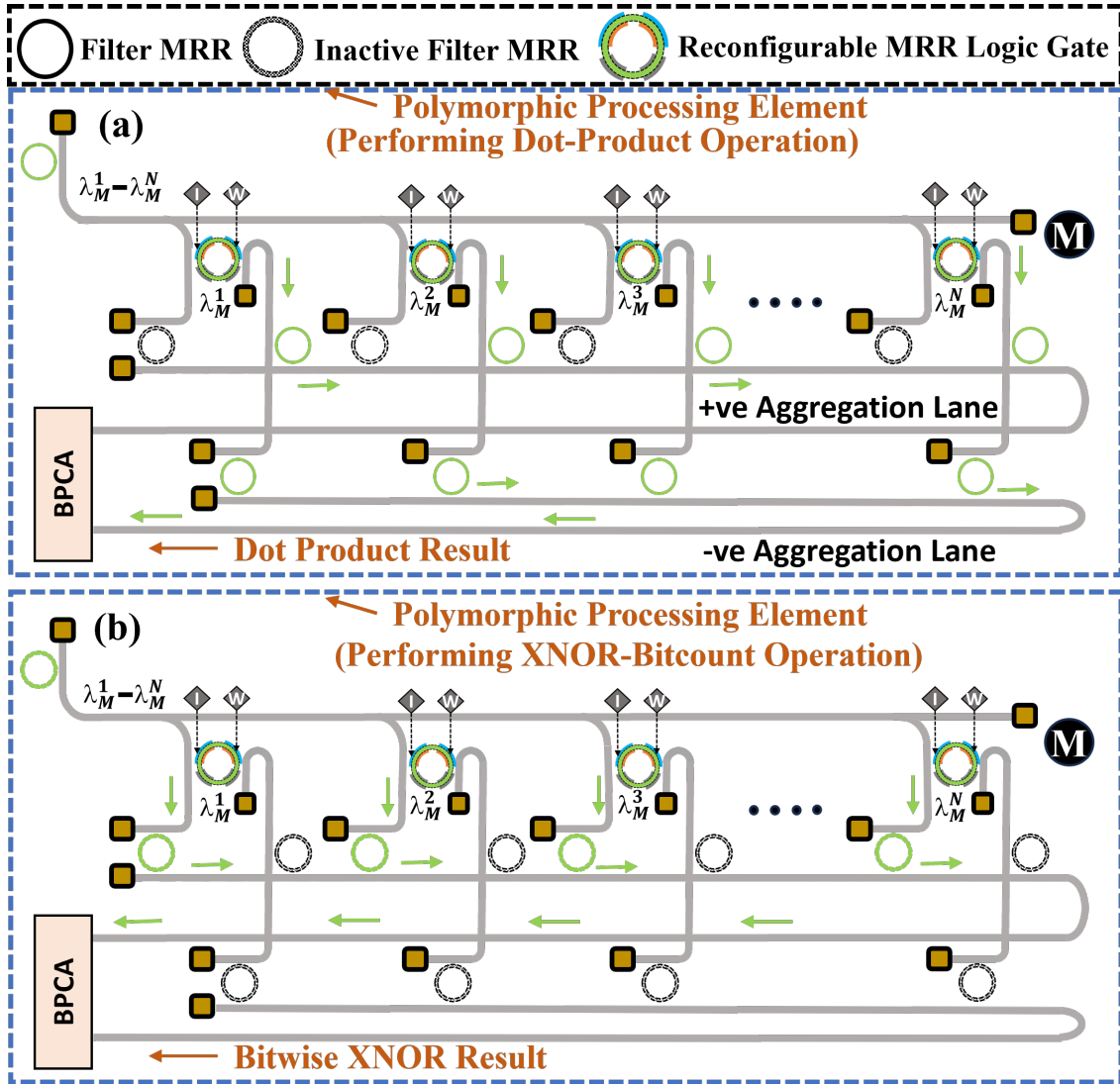


Figure 9.5: Operation of SCOAR PE (a) Dot product operation mode (b) XNOR-Bitcount operation mode.

trough ports. Each trough port waveguide is equipped with a MRR to drop the XNOR result onto +ve aggregation lane. In this mode, the MRR filters on the drop port waveguide are turned off. The +ve aggregation lanes guide the resultant bits of XNOR operation to BPCA for bit counting.

9.4.3 Reconfigurable Logic Gate

The design of our invented Reconfigurable Logic Gate (RLG) is illustrated in Fig. 9.4(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. Fig. 9.4(b) and Fig. 9.4(c) show the passbands of the MRR for differ-

ent operand inputs and temperature conditions. To operate as an XNOR gate, the MRR's temperature can be increased using the integrated microheater (Fig. 9.4(a)), to consequently tune its operand-independent resonance from its fabrication-defined initial position η to its programmed position κ (blue passband; Fig. 9.4(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. 9.4(b)). Based on the MRR resonance passband's programmed position κ relative to λ_{in} , the through-port transmission ($TT(\lambda_{in})$) of the MRR provides bit-wise logical XNOR operation between the input bits i and w . Similarly, the MRR can be reconfigured to operate as an AND logic gate at the drop port by changing the temperature to tune the resonance position κ (magenta passband; Fig. 9.4(c)). Then, for each bit combination at the operand terminals, the MRR's resonance passband electro-refractively moves to an operand-driven position (red and blue passbands in Fig. 9.4(c)). The MRR resonance passband's drop-port transmission ($DT(\lambda_{in})$) provides bit-wise logical AND operation between the input bits i and w . Thus, using the integrated microheater, our RLG can be programmed to operate as XNOR or AND logic gate based on the processing requirements. To assess the operational speed of our RLG, we conducted an analysis focusing on the maximum operating bitrate (BR) under specific conditions: FWHM=1.2nm, $P_{Laser} = 0\text{dBm}$, and OMA= -5dBm. Our findings indicate that the RLG can achieve a BR of 42 GS/s when configured as an AND gate, and a BR of 41 GS/s when configured as an XNOR gate. The microheater power and programmed detuning values for the RLG are provided in the inset table of Fig. 9.4(a).

To validate the operation of our RLG, we performed the transient analysis, as shown in Fig. 9.4(d). For that, we modeled and simulated our RLG using the foundry-validated tools from Ansys/Lumerical's DEVICE, CHARGE, and INTERCONNECT suites [6]. Fig. 9.4(c) shows two input bit-streams $I = \{i_1^1, i_2^1, \dots, i_8^1\}$ and $W = \{w_1^1, w_2^1, \dots, w_8^1\}$ applied to the two PN junctions of our RLG configured as XNOR gate and AND gate at a DR = 10 GS/s. By looking at the output optical trace $TT(\lambda_{in})$ and $DT(\lambda_{in})$ in Fig. 9.4 (c), we can say $TT(\lambda_{in}) = \{i_1^1 \odot w_1^1, \dots, i_8^1 \odot w_8^1\}$ and $DT(\lambda_{in}) = \{i_1^1 \cdot w_1^1, \dots, i_8^1 \cdot w_8^1\}$, which validates the functionality of our RLG as a logical XNOR gate and AND 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 0.011mm².

9.4.4 RLG Input Peripherals

Our RLG input peripherals consist of a binary-to-stochastic (B-to-S) number generator and serializers, as shown in Fig. 9.3(b). The B-to-S generator converts a binary input value I_b and binary weight value W_b into unipolar stochastic bit-streams I and W . Unlike prior work [186], we employ the deterministic B-to-S generator proposed in [190] for binary-to-stochastic conversion, which alleviates the need for high area and latency-incurring look-up tables. The B-to-S generator [190] produces two unipolar

stochastic bit-streams I and W by maintaining the marginal probability of one bit-stream (i.e., I or W) equal to its conditional probability given the other bit-stream (i.e., I given W or W given I) [196]. This method of B-to-S conversion ensures error-free stochastic multiplication [196]. The precision of the bitstreams depends on the length $L = 2^b$ of I and W . The length of the bitstreams is dynamically adapted by the B-to-S generator based on the precision requirements of the processing model. For example, to process a two-layer heterogeneously quantized CNN with 4-bit precision I_b and W_b for layer 1 and 8-bit precision I_b and W_b for layer 2, the deterministic B-to-S generator produces the stochastic bitstreams I and W of $L = 2^4 = 16$ during the processing of layer 1, and dynamically varies the length of the stochastic bitstreams I and W to $L = 2^8 = 256$ during the processing of layer 2 to achieve 8-bit precision. Thus, stochastic computing’s inherent flexibility allows us to set the precision by simply changing the length of the bit stream. Hence, changing precision levels across the layers of the heterogeneous quantized CNNs can be processed efficiently. Furthermore, these generated bitstreams I and W are fed to the RLG via high-speed serializers and drivers for stochastic multiplication. However, the length of the stochastic bit-streams increases exponentially with precision, and the stochastic representation of binary values above 8-bit binary precision incurs high latency and is detrimental to the throughput. Therefore, we limit the maximum supported precision of SCOAR PPE to 8-bits.

9.4.5 Balanced Photocharge Accumulator

From Section 9.4.1, the stochastic multiplication bit-streams or XNOR bits generated by RLGs are guided to a BPCA, where they are accumulated to generate a binary output value equivalent to the dot product result and in the case of BNNs, bit counting of XNOR bits is performed. Our BPCA is inspired by the time integrating receiver (TIR) design from [150] and the photodetector-based optical-pulse accumulator design from [29]. As shown in Fig 9.3(c), a BPCA circuit employs two photodiodes, each connected to the +ve aggregation lane and -ve aggregation lane of the PPE. These photodiodes are interlinked in a balanced configuration, commonly referred to as a balanced photodiode (BPD) configuration. The BPD is connected to a TIR via a switch (S_0). The TIR comprises an amplifier and a feedback capacitor/switch (S_1) pair (Fig. 9.3(c)). It functions as a current-to-voltage converter circuit by integrating the incoming differential electrical current over a period. The balanced photodetectors generate a differential current pulse for each optical logic ‘1’ incident upon it. This current pulse accumulates a certain amount of charge on the capacitor of the active TIR circuit (e.g., the circuit with a C1 capacitor); as a result, the capacitor accrues an analog voltage level. Hence, when one or more output optical bit-streams are incident upon the photodetectors, the total accumulated charge (and thus, the accrued analog voltage level) on the active capacitor (e.g., C1) is proportional to signed accumulation of the total number of ‘1’s in the incident bit-streams from +ve and -ve aggregation lanes. Prior works [186, 188] have used PDs and TIR configuration in this setup to perform signed accumulation and bit count operations using heterodyne incoherent superposition at PDs, however, none of the prior works

have explored the homodyne incoherent superposition of optical pulses at PDs to perform accumulation. For the first time, our BPCA employs homodyne incoherent superposition at PD which allows all the RLGs in a PPE to operate on a single wavelength λ_i . By designing the PPE to operate on a single wavelength λ_i , we remove FSR constraint on achievable N which improves the scalability of SCOAR PPE (more on this in Section 9.5). With homodyne incoherent superposition, the photodetectors accumulate optical pulses from all the RLGs. Depending on the operation mode, the output analog voltage computed corresponds to the unipolar unscaled addition [196] of the stochastic bit-streams or XNOR bitcount result. To convert this analog voltage into a binary value, the BPCA circuit employs an analog-to-digital converter (ADC). This binary value is the result of dot product operation for heterogeneous and homogeneous quantized CNN models and XNOR-bit count operation for BNNs.

9.5 Scalability Analysis

To determine the achievable size N for our SCOAR PU at maximum supported precision $b=8$, we adopt the scalability analysis equations (Eq. 9.1, Eq. 9.2, and Eq. 9.3) from [9] and [152, 186]. Table 9.1 reports the definitions of the parameters and their values used in these equations. our SCOAR PU processes stochastic bit-streams in dot product mode and binary values in XNOR-bit count mode, in both modes it requires the bit resolution of $B_{Res}=1$ -bit in the equations. In addition, we conservatively choose to operate RLGs at BR=30Gbps. As shown in Fig. 9.3(a), the M of the SCOAR PU is determined by the number of combs supported by the on-chip laser source, and we fix the $M=25$ with each comb providing a minimum laser power intensity of $P_{Laser} = -3\text{dbm}$ [135]. We first solve Eq. 9.1 and Eq. 5.4 for data rate (DR)=BR* 2^b , to find P_{PD-opt} to be -17.2 dBm. Then, we solve Eq. 5.4 for N , to find $N=1042$, which is a very large N compared to analog PUs that have $N \leq 44$ [152]. This large size of N is possible due to two reasons, firstly, SCOAR’s spectrally hitless PPE architecture (as discussed earlier in Section 9.4.1) due to which SCOAR significantly reduces the crosstalk-related power penalty contributing to $P_{penalty}$. Secondly, the prior works’ PEs utilize dense wavelength division multiplexing (DWDM), and the number of wavelengths multiplexed imposes a limit on the achievable N . This limit is defined by the available free spectral range (FSR) and channel spacing (CS) i.e $N=(\text{FSR}/\text{CS})$. Considering optimistic values of FSR=50nm and CS=0.25nm, the theoretical maximum achievable $N=200$. In contrast, SCOAR PE does not employ DWDM and uses a single wavelength in its PPE by exploiting homodyne incoherent superposition at a BPCA. By unlocking the potential of BPCA, SCOAR removes the FSR and CS-related limit on achievable N . These enable SCOAR to achieve higher N compared to prior analog accelerators. On the other hand, prior stochastic computing-based optical accelerator SCONNA [186] achieves $N=M=176$ which results in a total of $N \times M=30976$ parallel multiplication operations in a cycle, this is slightly higher than SCOAR’s $N \times M=1042 \times 25=26050$. However, SCONNA employs a huge number of on-chip laser diodes and look-up tables that incur high static power consumption, and latency which result in lower throughput and energy effi-

Table 9.1: Definition and values of various parameters used in Eq. 9.1, Eq. 9.2, and Eq. 9.3 (from [9, 152, 186]) for the scalability analysis.

Parameter	Definition	Value
P_{Laser}	Laser Power Intensity	-3 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
P_{EC-IL}	Fiber to Chip Coupling Insertion Loss	1.44
$P_{MRR-W-IL}$	Silicon Waveguide Insertion Loss	0.3 dB/mm
$P_{splitter-IL}$	Splitter Insertion Loss	0.01 dB
P_{MRM-IL}	Optical Microring Modulator Insertion Loss	4 dB
P_{MRR-IL}	Optical Microring Resonator Insertion Loss	0.01 dB
$P_{MRM-OBL}$	Out of Band Loss	0.01 dB
$P_{Penalty}$	SCOAR Network Penalty	1.8 dB

ciency compared to SCOAR (more details in section 9.6.3). In addition, SCONNA's usage of $N=176$ on-chip LD is far away from practical implementation.

$$B_{Res} = \frac{1}{6.02} \left[20 \log_{10} \left(\frac{R \times P_{PD-opt}}{\beta \sqrt{\frac{DR}{\sqrt{2}}}} - 1.76 \right) \right] \quad (9.1)$$

$$\beta = \sqrt{2q(RP_{PD-opt} + I_d) + \frac{4kT}{R_L} + R^2 P_{PD-opt}^2 RIN} + \sqrt{2qI_d + \frac{4kT}{R_L}} \quad (9.2)$$

$$\begin{aligned} P_{O/p}(dBm) = & P_{Laser} - P_{SMF-att} - P_{EC-IL} - P_{Si-att} \times N \times d_{MRR} \\ & - P_{MRM-IL} - (N-1)P_{MRM-OBL} - P_{splitter-IL} \times \log_2(M) \\ & - P_{MRR-W-IL} - (N-1)P_{MRR-W-OBL} - P_{penalty} - 10 \log_{10}(N) \end{aligned} \quad (9.3)$$

9.6 Evaluation

9.6.1 System Level Implementation of SCOAR

Fig. 9.6 illustrates the system-level implementation of our SCOAR accelerator. It consists of global memory that stores mixed precision model parameters and a pre-processing and mapping unit. It has a mesh network of tiles. Each tile contains 4 PUs

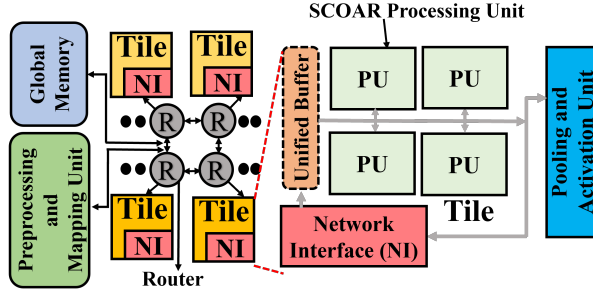


Figure 9.6: System-level overview of our SCOAR CNN accelerator.

interconnected (via H-tree) with a unified buffer, as well as pooling and activation units. Each PU consists of multiple PPEs and each PPE is equipped with a dedicated input and output FIFO buffer [192] to store intermittent weights, inputs, and $psum$ results.

9.6.2 Simulation Setup

For evaluation purposes, we modeled our SCOAR accelerator based on Fig. 9.6 using our custom-developed, transaction-level, event-driven Python-based simulator. We simulated the inference for a total of six CNN models, comprising three mixed-precision models, one homogeneously quantized model, and two binary neural networks. Specifically, for mixed-precision models, we included DenseNet121 [71], MobileNetV2 [69], and ResNet50 [65]. We determined the optimal layer configuration for these models using LQ-Net techniques [205]. Additionally, we utilized an 8-bit quantized GoogleNet model from [126] as the homogeneously quantized model. Furthermore, we evaluated binarized ResNet18[65] and VGG-small[149] from [205]. We evaluate frames-per-second (FPS) and FPS/W (energy efficiency).

We compared our SCONNA with the prior accelerators HQNNA [156], SCONNA [186], ROBIN [160], and OXBNN [188]. HQNNA is an analog optical accelerator designed for the efficient processing of heterogeneous quantized models that uses time division multiplexing-based bit slicing for supporting precision variation among the layers. SCONNA is a stochastic computing-based digital optical accelerator designed to perform inference of 8-bit quantized CNNs, it uses look-up tables for binary to stochastic conversion, and to enable processing of mixed precision models we integrated look-up tables corresponding to precision below 8-bits. ROBIN is an analog optical accelerator for BNNs, we employ bit slicing of weights and inputs to match the precision supported by ROBIN to process heterogeneous quantized CNNs. In addition to ROBIN, we consider another BNN-specific optical accelerator that employs single MRR-based XNOR gates [188] to process BNNs. Each accelerator variant is evaluated for the output stationary (OS) dataflow. Both HQNNA and ROBIN operate at 5 GS/s whereas SCONNA is operated at 30 Gbps. In addition, HQNNA has dedicated PU units to PU_Conv to process the convolution layers and PU_FC for fully connected layers. We consider two variants of ROBIN: ROBIN Energy-Optimized

Table 9.2: Accelerator Peripherals and DPU Parameters [152]

	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
DAC (ALL)¹ [181]	12.5	0.78ns	2.50E-3
DAC(SCOAR)²[82]	26	0.78ns	6.00E-3
EO Tuning	80 μ W/FSR	20ns	-
TO Tuning	275 mW/FSR	4 μ s	-

(ROBIN_EO) and ROBIN Performance-Optimized (ROBIN_PO)[160]. For a fair comparison, we performed area proportionate analysis, wherein we altered the PE count for each photonic accelerator across all of the accelerator’s PUs to match with the area of SCOAR having 50 PUs. Accordingly, the scaled PE counts of ROBIN_PO ($N=50$), ROBIN_EO ($N=10$), OXBNN ($N=53$), SCONNA ($N=176$) and HQNNA (PU_conv= $N=20$, PU_FC= $N=50$) are 65, 150, 270, 30, and 90 (PU_conv = 56 and PU_FC=34), respectively. Table 9.2 gives the parameters used for our evaluation.

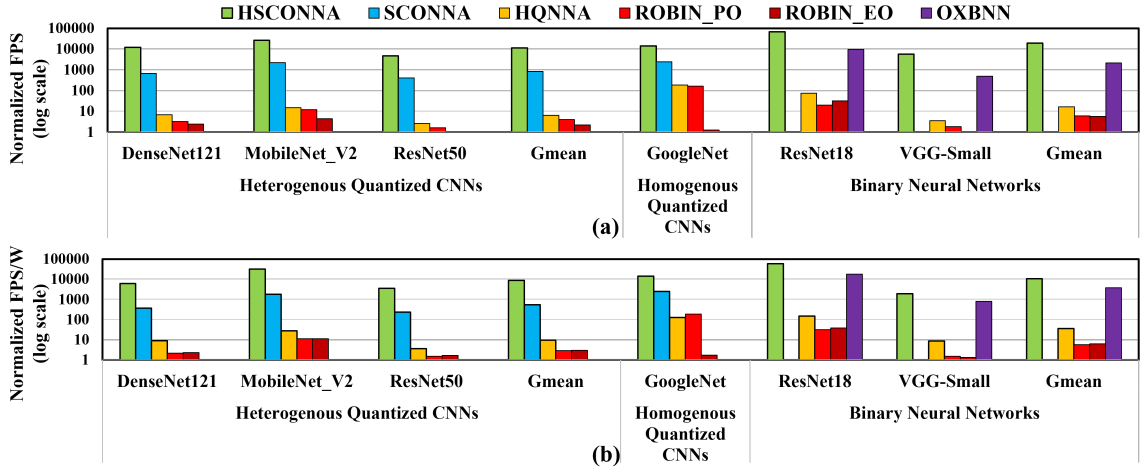


Figure 9.7: (a) Normalized FPS (log scale) (b) Normalized FPS/W (log scale) for SCOAR versus prior optical accelerators with input batch size=1. FPS and FPS/W results of heterogeneous CNNs are normalized relative to ROBIN_EO for ResNet50, and homogeneous CNNs are normalized relative to ROBIN_EO for VGG-small.

9.6.3 Evaluation Results

Fig. 9.7(a) compares the FPS values (log scale) achieved by each accelerator across various quantized CNNs. SCOAR significantly outperforms the analog optical accelerators HQNNA, ROBIN_PO and ROBIN_EO by $1758\times$, $2861\times$, and $5122\times$, respectively, on gmean across the heterogenous quantized CNNs. Similarly, SCOAR also achieves $76\times$, $85\times$, and $11313\times$ better FPS than analog optical accelerators HQNNA, ROBIN_PO and ROBIN_EO, respectively, for homogenous quantized CNNs. These benefits are mainly associated with the superior N , precision adaptive input peripherals, higher BR of SCOAR compared to the analog optical accelerators and in-situ accumulations at BPCA. When compared to analog accelerators our SCOAR with spectrally hitless architecture supports a larger PU size ($N=1042$), i.e., the count of multiplication operation N and the number of parallel multiplication operations M ($=25$), which increases the overall throughput with improved parallelism. Secondly, SCOAR, with its utilization of stochastic computing and dynamic precision adaptive input peripherals, eliminates the latency incurred with bit slicing in ROBIN and shift and accumulation in HQNNA. Third, the higher operating BR=30Gbps compensates for the lengthy stochastic bit-streams of 2^B bits used by SCOAR. Fourth, our BPCA eliminates the latency corresponding to $psum$ reductions. Furthermore, SCOAR also improves the throughput of BNN inference, SCOAR achieves at least $3212\times$, $3455\times$, and $9\times$ better FPS than prior optical BNN accelerators ROBIN_PO, ROBIN_EO, and OXBNN. SCOAR also achieves $1200\times$ better throughput than HQNNA on BNNs. Most of these benefits stem from the advantages discussed earlier; however, it’s important to note that the gains of SCOAR are less when compared to OXBNN. This is due to the fact that OXBNN also conducts in-situ accumulations and operates at 30Gbps. Furthermore, SCOAR performs exceptionally well against its stochastic computing counterpart SCONNA, achieving $13\times$ better FPS on gmean across heterogeneous models and $6\times$ better FPS on homogeneous models than SCONNA. SCOAR utilizes significantly faster and more compact deterministic binary-to-stochastic number generators instead of the bulky and time-consuming lookup tables used in SCONNA. Moreover, SCONNA is not capable of running BNNs due to its functionally fixed PEs, which cannot perform XNOR operations. Overall, SCOAR gives exceedingly better FPS compared to the prior works.

Fig. 9.7(b) gives the energy efficiency (FPS/W) values for each accelerator across heterogenous precision, homogenous precision, and binary neural network models. It is evident that SCOAR attains substantially better energy efficiency than prior optical accelerators. Our SCOAR gains $900\times$, $3100\times$, and $2900\times$ better FPS/W against HQNNA, ROBIN_PO, and ROBIN_EO respectively, on gmean across the heterogenous quantized models. SCOAR also attains significantly improved energy efficiency for homogenous quantized and BNN models. As shown in the Fig. 9.7(b), SCOAR achieves $297\times$, $1883\times$, $1721\times$, and $2.8\times$ better energy efficiency compared to HQNNA, ROBIN_PO, ROBIN_EO, and OXBNN, respectively on gmean, across two BNN models. These energy efficiency benefits are due to the improved throughput and flexible precision support of SCOAR PUs. In addition, due to BPCA, SCOAR performs in-situ accumulation, leading to significant energy savings by drastically

Table 9.3: Top-1 and Top-5 inference accuracy comparison of various binary to stochastic conversion techniques for 8-bit quantized CNNs GoogleNet and ResNet50 and ImageNet dataset [45].

	GoogleNet		ResNet50		Gmean	
Accuracy Drop (%)	Top1	Top5	Top1	Top5	Top1	Top5
UGEMM	1.2	0.9	0.9	1.8	1	1.2
Gaines	1.7	2	1.6	2.5	1.6	2.2
Jenson	1.4	1.7	1.4	2	1.4	1.8
SCOAR	0.5	0.6	0.3	0.5	0.4	0.5

reducing the frequency of analog-to-digital converter (ADC) usage. Furthermore, BPCA eliminates the need for a *psum* reduction network and its corresponding energy consumption. Moreover, leveraging stochastic computing, SCOAR eliminates the necessity for energy-intensive shifters and adders used in the time division multiplexing scheme of HQNNA. As discussed in the preceding section, by replacing lookup tables with deterministic binary-to-stochastic number generators, SCOAR reduces static power consumption and read access energy compared to SCONNA, resulting in at least $16\times$ better energy efficiency for SCOAR. Overall, SCOAR significantly improves the throughput and energy efficiency compared to the tested analog and digital optical accelerators.

The area efficiency values (FPS/W/mm²) for each accelerator across various CNNs are similar to the energy efficiency (FPS/W) values for area proportional analysis (when the area of all the accelerators is matched to the area of SCOAR). Therefore, we have not reported area efficiency results. Overall, SCOAR significantly improves throughput (FPS) and energy efficiency (FPS/W) across various quantized models compared to the tested optical accelerators.

9.6.4 Inference Accuracy Results

As discussed in Section 9.4.4, the binary to stochastic conversion can incur errors in the computed stochastic multiplication results. To evaluate the impact of the stochastic to binary (B-to-S) conversion technique used in our SOAR on the CNN inference accuracy, we simulated the inference of GoogleNet[162] and ResNet50[65] with our deterministic B-to-S conversion and compared it against prior B-to-S techniques like UGEMM[196], Gaines[54], and Jenson[78]. We integrated our custom simulator with ML-framework PyTorch [126] and performed the inference using ImageNet validation dataset [45] (50k images and 1k classes). Table 9.3 reports the Top-1 and Top-5 inference accuracy drop obtained for our SCOAR and prior techniques for 8-bit integer-quantized GoogleNet[162] and ResNet50[65]. As evident, SCOAR yields the least accuracy drop of 0.4% and 0.5% better Top-1 and Top-5 accuracy, respectively, on gmean across the GoogleNet[162] and ResNet50[65] CNNs. Moreover, our SCOAR accelerator’s significant gains in the FPS, FPS/W, and FPS/W/mm², overshadow the minor drop in the CNN inference accuracy.

9.7 Summary

In this paper, we presented SCOAR, a novel MRR-based CNN accelerator. The processing element of our SCOAR employs a spectrally hitless array of novel reconfigurable logic gates that efficiently meet the computation requirements of homogeneous, heterogeneous, and binary quantized CNNs. In addition, SCOAR exploits the homodyne incoherent superposition of the balanced photocharge accumulator to remove the FSR-imposed scalability limit and perform in-situ temporal accumulations. Our evaluation results for six quantized CNN models show that SCOAR provides improvements of at least $1758\times$ ($900\times$), $76\times$ ($6\times$), and $9\times$ ($2.8\times$) better in throughput(energy efficiency) for heterogeneous quantized, homogeneous quantized, and binary quantized CNN, respectively, when compared to four optical accelerators HQNNA, SCONNA, ROBIN and OXBNN, with Top-1 and Top-5 accuracy drop of only up to 0.4% and 0.5% for homogenous quantized CNNs. Our evaluation results for six quantized CNN models show that SCOAR provides improvements of at least $1758\times$ ($900\times$), $76\times$ ($6\times$), and $9\times$ ($2.8\times$) better in throughput(energy efficiency) for heterogeneous quantized, homogeneous quantized, and binary quantized CNNs, respectively, when compared to four optical accelerators HQNNA, SCONNA, ROBIN and OXBNN, with Top-1 and Top-5 accuracy drop of only up to 0.4% and 0.5% for homogenous quantized CNNs.

Chapter 10 Conclusions and Future Work

In this report, we presented several solutions to address various design challenges encountered by PIC-based AI accelerator systems. A recap of each of our contributions is discussed in the upcoming paragraphs.

In our first contribution in Chapter 2, we introduced a silicon MR-based chip-scale accelerator for delayed feedback reservoir computing (DFRC), enabling on-chip integration and significantly reducing training times. Leveraging the rich nonlinearity of the active MRR, we implemented the nonlinear node in the reservoir layer of the accelerator. Then, by utilizing the photonic waveguide as the feedback delay loop, we enabled full on-chip integration of the reservoir layer. Our evaluation with benchmark tasks showed that our MRR-based DFRC accelerator achieves 35% and 98.7% lower NRMSE, up to 58.8% less average SER, and up to 93x faster training time compared to a photonic DFRC accelerator from prior work.

In our second contribution in Chapter 3, we presented an insertion loss-aware framework *PROTEUS* for co-optimizing the laser power consumption and performance of emerging PNoCs. *PROTEUS* dynamically adapts the key photonic link configuration parameters, such as Q -factor of microrings and bitrate of photonic data signals, to statically reduce the laser power consumption and opportunistically improve the packet transfer latency in PNoCs. *PROTEUS* exploits the dependence of BER power penalty in PNoCs on Q -factor and bitrate to balance the reduction in laser power consumption in PNoCs with the achieved aggregated datarate and packet latency. Our *PROTEUS* framework achieves up to 24.5% less laser power consumption, up to 31% less average packet latency, and up to 20% less energy-per-bit, compared to two other laser power management techniques from prior work.

In our third contribution in Chapter 4, we tackle the reconfigurability challenges in MRR-based CNN accelerators by introducing novel reconfigurable DPEs. These DPEs enable dynamic maximization of the size compatibility between the DPEs and the CNN tensors processed using them. Our reconfigurable DPEs seamlessly integrate with existing MRR-based CNN accelerators. Evaluation results demonstrated that our integrated accelerators with reconfigurable DPEs achieve a better balance between hardware utilization and CNN processing latency, leading to substantial improvements in Frames per second (FPS)(throughput) and FPS/W(energy efficiency).

In our fourth contribution in Chapter 5, we introduced a single-MRR-based optical XNOR gate (OXG) and a novel bitcount circuit called Photo-Charge Accumulator (PCA). Utilizing OXGs and PCAs, we developed a novel accelerator named OXBNN for processing BNN inferences. Our OXGs efficiently perform the necessary XNOR operations in BNNs using half the number of MRRs, thereby reducing both area and optical losses. Furthermore, our PCA enabled in-situ bitcount operations by leveraging incoherent heterodyne superposition at photodetectors and time-integrating receivers, thus eliminating the need for external electronic bit-count circuits. The combined advantages of OXGs and PCAs allow OXBNN to achieve FPS (throughput) and FPS/W(energy efficiency) improvements of up to 62 \times and

7.6 \times , respectively, compared to two state-of-the-art photonic BNN accelerators from previous works.

In our fifth contribution in Chapter 6, we addressed the significant scalability versus bit-precision trade-off inherent in analog optical CNN accelerators by integrating stochastic computing with MRR-based CNN accelerators. We developed an MRR-based optical stochastic multiplier (OSM) for stochastic multiplications and utilized multiple OSMs to create a novel stochastic computing-based CNN accelerator named SCONNA. The OSM was modeled using photonics foundry-validated simulation tools from ANSYS/Lumerical. By overcoming this tradeoff, SCONNA enhances the scalability of an MRR-based CNN accelerator by approximately 4 times. Our evaluation results for four CNN models demonstrate that SCONNA achieves improvements of up to 66.5 \times , 90 \times , and 91 \times in throughput, energy efficiency, and area efficiency, respectively, compared to two analog optical accelerators, with a negligible Top-1 accuracy drop of up to 0.4% for large CNNs and up to 1.5% for small CNNs.

In our sixth contribution in Chapter 7, we introduced a systematic approach for classifying prior incoherent MRR-based CNN accelerators based on the aggregation routing used by photonic interconnect organizations in their DPEs into Modulation-Aggregation-Splitting-Weighting (MASW), Aggregation-Splitting-Modulation Weighting, and Splitting-Modulation-Weighting-Aggregation (SMWA). We performed a comprehensive circuit-level comparative analysis of MASW, ASMW, and SMWA organizations and identified that each organization incurs different magnitudes of crosstalk noise and optical signal losses. We demonstrated that due to these variations, each organization achieves different levels of processing parallelism. At the system level, our evaluation results for four CNN models demonstrated that SMWA organization achieves up to 4.4 \times , 5 \times , and 5.2 \times better throughput, energy efficiency, and area-energy efficiency, respectively, compared to ASMW and MASW organizations on average.

In our seventh contribution in Chapter 8, we presented HEANA, a novel hybrid time-amplitude analog optical GEMM accelerator. The dot product element of our HEANA employs a spectrally hitless array of novel hybrid time-amplitude analog optical modulators (TAOM) that perform multiplication operations, combined with a balanced photo-charge accumulator (BPCA) that performs spatiotemporal accumulation of the individual multiplication results from TAOMs. Each TAOM utilizes a single active microring resonator (MRR) to perform multiplication operations, allowing for the electro-optic tuning of both input and weight values. This enables the flexibility of executing various CNN dataflows namely input stationary, output stationary, and weight stationary dataflows. Moreover, the spectrally hitless arrangement of TAOMs in HEANA eliminates spectral interference and various crosstalk effects encountered by current MRR-enabled analog photonic accelerators. In addition, the single MRR implementation of TAOMs substantially reduces the area consumption, implementation complexity, and insertion losses in HEANA, thereby increasing its energy efficiency. Also, the BPCA circuit in HEANA enables in-situ accumulation of a large number of partial sums, thereby reducing the overall latency and energy consumption of CNN processing. We performed detailed modeling and characterization of our TAOM unit using photonics foundry-validated tools from ANSYS/Lumerical.

Here, we performed time-domain simulations from which we evaluated the performance of our TAOM in terms of accuracy and precision. We have also performed a scalability analysis of our HEANA’s dot product units, to determine their achievable maximum size and supported bit-precision. Finally, we evaluated HEANA for input stationary, output stationary, and weight stationary dataflows at the system-level, and compared its performance with two well-known photonic CNN accelerators from prior works. Our system-level evaluation results for four CNN models show that HEANA provides improvements of up to $25\times$ and $32\times$ in throughput, and energy efficiency, respectively, compared to two prior optical analog accelerators AMM and MAM, with Top-1 accuracy drop of only up to 0.1%.

In our eighth contribution in Chapter 9, we presented SCOAR, a novel MRR-based CNN accelerator. The processing element of our SCOAR employs a spectrally hitless array of novel reconfigurable logic gates that dynamically adapt to the computation requirements such as variable precision and operations like XNOR for homogeneous, heterogeneous, and binary quantized CNNs. SCOAR also exploits the homodyne incoherent superposition of the balanced photo charge accumulator to eliminate the FSR-imposed scalability limit and perform in-situ temporal accumulations. Our system-level evaluation results show that SCOAR provides improvements of at least $1758\times$ ($900\times$), $76\times$ ($6\times$), and $9\times$ ($2.8\times$) better in throughput (energy efficiency) for heterogeneous quantized, homogeneous quantized, and binary quantized CNNs, respectively, compared to four optical accelerators, with Top-1 and Top-5 accuracy drops of only up to 0.4% and 0.5% for homogeneous quantized CNNs.

10.1 Future Work

As AI technologies continue to advance, the complexity of tasks they are expected to perform is escalating rapidly. This growth is fueled by the need for more sophisticated algorithms, larger datasets, and higher precision models. Consequently, AI computing requirements are skyrocketing, necessitating more powerful hardware, software, and efficient computational architectures to keep pace with the evolving demands. As the AI computing demands keep scaling, photonic integrated circuits-based AI accelerators will continue to face new design challenges. Taking this into consideration, we provide the following directions for future research.

Analysis of Optical Loss and Crosstalk Noise in MRR-based Incoherent Photonic Accelerators

In Chapter 6, we presented a comparative analysis of the impact of DPU organization on scalability, considering variations in optical losses. However, the MRR also encounters inherent optical losses and crosstalk noise arising from fabrication imperfections and undesired optical couplings, which become increasingly significant as the network scales up. Consequently, the inference accuracy of MRR-based incoherent photonic accelerators can be compromised by such inefficiencies. While prior works [152, 157, 9], including Chapter 6, have estimated these losses at a conceptual level, we propose the development of comprehensive models to accurately calculate losses

from the device to the system level. These proposed models can be applied to any MRR-based incoherent photonic accelerator architecture with various configurations to analyze the effects of loss and crosstalk. Such an analysis is crucial for meeting inference accuracy and scalability requirements when designing an MRR-based incoherent photonic accelerator.

A Spatio Temporal In-Situ accumulator for efficient partial sum reductions in MRR based Photonic CNN accelerators

Prior work [95, 120] has demonstrated that inter-DPU and intra-DPU communication in MRR-based Photonic CNN accelerators significantly contributes to latency and power consumption, thereby affecting throughput and energy efficiency. Communication requirements primarily arise during the processing of partial sum accumulations. Current accelerators typically rely on electronic reduction tree-based networks such as S_TREE [81], S_TREE_ACC [93], and STIFT [120] for reduction, which involves ADC conversions, buffer accesses, and communication overhead. Chapters 6 and 7 of our work addressed the lack of photonic in-situ spatio-temporal accumulators by introducing a balanced photocharge accumulator (BPCA). However, these chapters and prior works lack a comprehensive analysis of applying BPCA under various clustering configurations at DPU-level and DPE-level and data mapping with different dataflows for available MRR-based DPU architectures. Our goal in this work is to conduct such an analysis to demonstrate the advantages of BPCA over electronic reduction networks. Furthermore, we aim to identify optimal configuration parameters such as clustering type (intra-DPE versus intra-DPU), clustering size, and dataflow to maximize the throughput and energy efficiency of various MRR-based DPU accelerator architectures.

A DWDM enabled Microring resonator based delay feedback reservoir

As demonstrated in Chapter 2, MRR-based Delay Feedback Reservoir Computing (DFRC) has shown comparable performance to its electronic and photonic counterparts. However, both the architecture proposed in Chapter 2 and previous works are limited to performing training or inference on a single task, whereas modern AI workloads increasingly demand parallel processing capabilities. Thus, there is a pressing need to explore opportunities for parallel processing in PIC-based DFRC accelerators. To address the lack of concurrency in running parallel model training and inference on PIC-based DFRC accelerators, we propose leveraging the dense wavelength division compatibility of MRRs. Our plan involves integrating more than one MRR in the DFRC loop, with each MRR handling an individual task while sharing the same feedback loop to act as a reservoir. Furthermore, the evaluation of PIC-based DFRC accelerators in the literature has primarily focused on conventional RC Benchmarks, lacking analysis of recent RC tasks and comparisons with state-of-the-art RNN models. Therefore, we aim to fill this gap by conducting thorough evaluations of PIC-based DFRC accelerators on contemporary RC tasks and comparing their performance with state-of-the-art RNN models.

A case for Heterogeneous Architectures of Electro-photonic GEMM Accelerators

It has been shown that electro-photonic (EP) GEMM accelerators can be employed for high-performance as well as embedded/edge computing applications [150]. However, the method of operation for an EP GEMM accelerator has to differ for the high-performance application compared to the applications at the edge. This is because, for the application at the edge, energy efficiency is typically prioritized over throughput. Therefore, for the application at the edge, energy-friendly data encoding/signaling and other operating conditions are utilized, whereas, for the high-performance application, throughput-friendly signaling and operating conditions are utilized. Consequently, the use of binary/unary signaling (analog or mixed signaling) is common for the at-edge (high-performance) application [152, 150]. However, to minimize energy consumption at a BPD/BPCA, it is common to select a very slow sampling rate of the BPD/BPCA for the application at the edge [150, 188]. This necessitates that the BPD/BPCA allows for accumulations (signed sum-of-products) to be performed both temporally as well as spatially to allow multiple fast-arriving product values to be accumulated every sampling period. In contrast, a fast sampling rate is required at BPDs/BPCAs for the high-performance application, which makes it sufficient to enable spatial-only accumulations at the BPDs/BPCAs. To provide efficient software-level support for this kind of heterogeneity of operating conditions and applications, there is an opportunity to develop new languages and compiler tools.

Appendix

- <https://github.com/uky-UCAT/Photonic-Interconnects-Simulator> (A cycle-accurate C++ photonic interconnect simulator derived from NOXIM and GEM5 as part of Chapter 3. The repository contains the implementation of the framework proposed in Chapter 3 and the corresponding simulation files.)
- <https://github.com/uky-UCAT/CASES2022> (The python scripts developed to perform scalability analysis of analog MRR-based CNN accelerators as part of Chapter 4.)
- https://github.com/uky-UCAT/B_ONN_SIM (A python-based simulator developed as part of Chapter 5. The simulator can evaluate the BNN inference on various MRR-based BNN accelerators to determine metrics like throughput, energy efficiency, and area efficiency.)
- https://github.com/uky-UCAT/SC_ONN_SIM (A transaction-level, event-driven python-based simulator developed as part of Chapter 6. It serves as a tool for the performance evaluation of stochastic computing-based optical neural network accelerators designed for various quantized Convolutional Neural Network models.)

Bibliography

- [1] IVC102 data sheet, product information and support — TI.com — ti.com. <https://www.ti.com/product/IVC102>. [Accessed 17-10-2023].
- [2] Multisim. <https://www.ni.com/en-us/shop/software/products/multisim.html>.
- [3] Pytorch 2.0 documentation — pytorch.org. <https://pytorch.org/docs/stable/generated/torch.nn.Unfold.html>.
- [4] The santa fe atime series competition data. http://www.comp-engine.org/timeseries/time-series_data_source/source-151/.
- [5] *Recurrent Neural Networks Architectures*, chapter 5, pages 69–89. John Wiley and Sons, Ltd, 2001.
- [6] Pic design and simulation software - lumerical interconnect. <https://www.lumerical.com/products/interconnect/>, Apr 2023.
- [7] A. H. Ahmed, A. Sharkia, B. Casper, S. Mirabbasi, and S. Shekhar. Silicon-photonics microring links for datacenters—challenges and opportunities. *IEEE Journal of Selected Topics in Quantum Electronics*, 22(6):194–203, 2016.
- [8] F. Akopyan et al. Truenorth: Design and tool flow of a 65 mw 1 million neuron programmable neurosynaptic chip. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 34(10):1537–1557, 2015.
- [9] M. A. Al-Qadasi, L. Chrostowski, B. J. Shastri, and S. Shekhar. Scaling up silicon photonic-based accelerators: Challenges and opportunities. *APL Photonics*, 7(2):020902, 2022.
- [10] A. Alaghi and J. P. Hayes. Survey of stochastic computing. *ACM Trans. Embed. Comput. Syst.*, 2013.
- [11] P. Antonik, M. Hermans, F. Duport, M. Haelterman, and S. Massar. Towards pattern generation and chaotic series prediction with photonic reservoir computers. In B. Jalali, S. K. Turitsyn, D. R. Solli, and J. M. Dudley, editors, *Real-time Measurements, Rogue Events, and Emerging Applications*, volume 9732, pages 21 – 32. International Society for Optics and Photonics, SPIE, 2016.
- [12] L. Appeltant, G. V. der Sande, J. Danckaert, and I. Fischer. Constructing optimized binary masks for reservoir computing with delay systems. *Scientific Reports*, 4(1), Jan. 2014.
- [13] L. Appeltant, M. Soriano, G. V. der Sande, J. Danckaert, S. Massar, J. Dambre, B. Schrauwen, C. Mirasso, and I. Fischer. Information processing using a single dynamical node as complex system. *Nature Communications*, 2(1), Sept. 2011.

- [14] L. Appeltant, M. Soriano, G. V. der Sande, J. Danckaert, S. Massar, J. Dambre, B. Schrauwen, C. Mirasso, and I. Fischer. Information processing using a single dynamical node as complex system. *Nature Communications*, 2(1), Sept. 2011.
- [15] A. Armin et al. The promise and challenge of stochastic computing. *TCAD*, 2018.
- [16] H. Bagherian et al. On-chip optical convolutional neural networks. *CoRR*, 2018.
- [17] 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*, 34(17):4043–4052, 2016.
- [18] M. Bahadori, S. Rumley, D. Nikolova, and K. Bergman. Comprehensive design space exploration of silicon photonic interconnects. *Journal of Lightwave Technology*, 34(12):2975–2987, 2016.
- [19] M. Bahadori, S. Rumley, R. Polster, A. Gazman, M. Traverso, M. Webster, K. Patel, and K. Bergman. Energy-performance optimized design of silicon photonic interconnection networks for high-performance computing. In *Design, Automation Test in Europe Conference Exhibition (DATE), 2017*, pages 326–331, 2017.
- [20] L. Baischer, M. Wess, and N. TaheriNejad. Learning on hardware: A tutorial on neural network accelerators and co-processors, 2021.
- [21] R. Balasubramonian, A. B. Kahng, N. Muralimanohar, A. Shafiee, and V. Srinivas. Cacti 7: New tools for interconnect exploration in innovative off-chip memories. *ACM Trans. Archit. Code Optim.*, 2017.
- [22] 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*, 26(1):1–13, 2020.
- [23] J. Bashir and S. R. Sarangi. Predict, share, and recycle your way to low-power nanophotonic networks. *J. Emerg. Technol. Comput. Syst.*, 16(1), oct 2019.
- [24] H. Benmezziane et al. A comprehensive survey on hardware-aware neural architecture search. *arXiv preprint arXiv:2101.09336*, 2021.
- [25] C. Bienia, S. Kumar, J. P. Singh, and K. Li. The parsec benchmark suite: Characterization and architectural implications. In *2008 International Conference on Parallel Architectures and Compilation Techniques (PACT)*, pages 72–81, 2008.

- [26] N. Binkert, B. Beckmann, G. Black, S. K. Reinhardt, A. Saidi, A. Basu, J. Hestness, D. R. Hower, T. Krishna, S. Sardashti, R. Sen, K. Sewell, M. Shoaib, N. Vaish, M. D. Hill, and D. A. Wood. The gem5 simulator. *ACM SIGARCH Computer Architecture News*, 39(2):1–7, May 2011.
- [27] L. S. Blackford et al. An updated set of basic linear algebra subprograms (blas). *ACM Transactions on Mathematical Software*, 2002.
- [28] 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. Silicon microring resonators. *Laser Photonics Reviews*, 6(1):47–73, 2012.
- [29] F. Brücknerhoff-Plückelmann et al. A large scale photonic matrix processor enabled by charge accumulation. *Nanophotonics*, 2022.
- [30] Cerebras. <https://www.cerebras.net/product-chip/>.
- [31] V. Chandra and R. Ranjan. Analysis of propagation loss in silicon-on-insulator based photonic rib waveguide with small cross section. In *2019 URSI Asia-Pacific Radio Science Conference (AP-RASC)*, pages 1–3, 2019.
- [32] C. Chen, J. L. Abellán, and A. Joshi. Managing laser power in silicon-photonic noc through cache and noc reconfiguration. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 34(6):972–985, 2015.
- [33] C. Chen and A. Joshi. Runtime management of laser power in silicon-photonic multibus noc architecture. *IEEE Journal of Selected Topics in Quantum Electronics*, 19(2):3700713–3700713, 2013.
- [34] Q. Cheng, J. Kwon, M. Glick, M. Bahadori, L. P. Carloni, and K. Bergman. Silicon photonics codesign for deep learning. *Proceedings of the IEEE*, 108(8):1261–1282, 2020.
- [35] S. V. R. Chittamuru, S. Desai, and S. Pasricha. SWIFTNoC. *ACM Journal on Emerging Technologies in Computing Systems*, 13(4):1–27, Aug. 2017.
- [36] F. Chollet. Xception: Deep learning with depthwise separable convolutions. *CoRR*, 2016.
- [37] F. Chollet et al. Keras. <https://keras.io/api/applications/>, 2015.
- [38] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014.
- [39] F. D.-L. Coarer, M. Sciamanna, A. Katumba, M. Freiburger, J. Dambre, P. Bienstman, and D. Rontani. All-optical reservoir computing on a photonic chip using silicon-based ring resonators. *IEEE Journal of Selected Topics in Quantum Electronics*, 24(6):1–8, 2018.

- [40] 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.
- [41] G. Dabos, G. Mourgias-Alexandris, A. Totovic, M. Kirtas, N. Passalis, A. Tefas, and N. Pleros. End-to-end deep learning with neuromorphic photonics. In *Integrated Optics: Devices, Materials, and Technologies XXV*, volume 11689, pages 56–66. SPIE, 2021.
- [42] S. Dave et al. Hardware acceleration of sparse and irregular tensor computations of ml models: A survey and insights. *Proceedings of the IEEE*, 2021.
- [43] L. De Marinis et al. Photonic neural networks: A survey. *IEEE Access*, 2019.
- [44] 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-photonics system for accelerating deep neural networks. *ArXiv*, abs/2109.01126, 2021.
- [45] J. Deng et al. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009.
- [46] D. Dhang, S. A. Hasnain, and R. Mahapatra. Mrec: A multilayer photonic reservoir computing architecture. In *20th International Symposium on Quality Electronic Design (ISQED)*, pages 170–175, 2019.
- [47] S. Dong, P. Wang, and K. Abbas. A survey on deep learning and its applications. *Comput. Sci. Rev.*, 40(C), may 2021.
- [48] F. Duport, A. Smerieri, A. Akrout, M. Haelterman, and S. Massar. Fully analogue photonic reservoir computer. *Scientific Reports*, 6(1), Mar. 2016.
- [49] H. El-Derhalli et al. Towards all-optical stochastic computing using photonic crystal nanocavities. *JETC*, 2021.
- [50] M. Fatica. Cuda toolkit and libraries. In *2008 IEEE Hot Chips 20 Symposium HCS*, pages 1–22, 2008.
- [51] J. Feldmann et al. Parallel convolutional processing using an integrated photonic tensor core. *Nature*, 2021.
- [52] T. Ferreira de Lima, E. Doris, S. Bilodeau, W. Zhang, A. Jha, H.-T. Peng, E. Blow, C. Huang, A. Tait, B. Shastri, and P. Prucnal. Design automation of photonic resonator weights. *Nanophotonics*, 11, 04 2022.
- [53] M. A. A. Fiers, T. Van Vaerenbergh, F. Wyffels, D. Verstraeten, B. Schrauwen, J. Dambre, and P. Bienstman. Nanophotonic reservoir computing with photonic crystal cavities to generate periodic patterns. *IEEE Transactions on Neural Networks and Learning Systems*, 25(2):344–355, 2014.

- [54] B. R. Gaines. *Stochastic Computing Systems*, pages 37–172. Springer US, Boston, MA, 1969.
- [55] P. Gao, Z. Liu, and X. Gui. Modeling and design of a 0.8–30 ghz tunable inductor-less divide-by-2 frequency divider with digital frequency calibration. In *2015 IEEE 58th International Midwest Symposium on Circuits and Systems (MWSCAS)*, pages 1–4, 2015.
- [56] Y. Gong, L. Liu, M. Yang, and L. D. Bourdev. Compressing deep convolutional networks using vector quantization. *ArXiv*, abs/1412.6115, 2014.
- [57] Graphcore. <https://www.graphcore.ai/products/ipu>.
- [58] J. Gu, C. Feng, Z. Zhao, Z. Ying, M. Liu, R. T. Chen, and D. Z. Pan. Squeeze-light: Towards scalable optical neural networks with multi-operand ring resonators. In *2021 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pages 238–243, 2021.
- [59] K. Guo, W. Li, and K. Zhong. Neural network accelerator comparison. <https://nicsefc.ee.tsinghua.edu.cn/project.html>.
- [60] M. Guo et al. A 29mw 5gs/s time-interleaved sar adc achieving 48.5db snr with fully-digital timing-skew calibration based on digital-mixing. In *VLSIC*, 2019.
- [61] 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*, ICML’15, page 1737–1746. JMLR.org, 2015.
- [62] C. Haffner, W. Heni, Y. Fedoryshyn, J. Niegemann, A. Melikyan, D. L. Elder, B. Baeuerle, Y. Salamin, A. Josten, U. Koch, C. Hoessbacher, F. Ducry, L. Juchli, A. Emboras, D. Hillerkuss, M. Kohl, L. R. Dalton, C. Hafner, and J. Leuthold. All-plasmonic mach–zehnder modulator enabling optical high-speed communication at the microscale. *Nature Photonics*, 9(8):525–528, July 2015.
- [63] R. Hamerly et al. Large-scale optical neural networks based on photoelectric multiplication. *Phys. Rev. X*, May 2019.
- [64] B. Hammer and J. Steil. Tutorial: Perspectives on learning with rnns. 05 2002.
- [65] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.
- [66] R. Hendry, D. Nikolova, S. Rumley, N. Ophir, and K. Bergman. Physical layer analysis and modeling of silicon photonic wdm bus architectures. 01 2014.

- [67] M. Hermans, M. C. Soriano, J. Dambre, P. Bienstman, and I. Fischer. Photonic delay systems as machine learning implementations. *Journal of Machine Learning Research*, 16(64):2081–2097, 2015.
- [68] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.
- [69] A. G. Howard et al. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *CoRR*, 2017.
- [70] C. Huang et al. Demonstration of scalable microring weight bank control for large-scale photonic integrated circuits. *APL Photonics*, Apr. 2020.
- [71] G. Huang et al. Densely connected convolutional networks. *CoRR*, 2016.
- [72] T. W. Hughes, M. Minkov, Y. Shi, and S. Fan. Training of photonic neural networks through in situ backpropagation and gradient measurement. *Optica*, 5(7):864–871, Jul 2018.
- [73] Isgcal. Isgcal/siphsimulink: Simulink library for silicon photonics systems simulation.
- [74] B. Jacob et al. Quantization and training of neural networks for efficient integer-arithmetic-only inference. In *CVPR*, 2018.
- [75] H. Jaeger. The” echo state” approach to analyzing and training recurrent neural networks-with an erratum note’. *Bonn, Germany: German National Research Center for Information Technology GMD Technical Report*, 148, 01 2001.
- [76] H. Jaeger. Adaptive nonlinear system identification with echo state networks. In *Proceedings of the 15th International Conference on Neural Information Processing Systems, NIPS’02*, page 609–616, Cambridge, MA, USA, 2002. MIT Press.
- [77] H. Jaeger and H. Haas. Harnessing nonlinearity: Predicting chaotic systems and saving energy in wireless communication. *Science*, 304(5667):78–80, 2004.
- [78] D. Jenson et al. A deterministic approach to stochastic computation. In *ICCAD*, 2016.
- [79] Z. Jia et al. Dissecting the graphcore IPU architecture via microbenchmarking. *CoRR*, 2019.
- [80] A. Joshi, C. Batten, Y.-J. Kwon, S. Beamer, I. Shamim, K. Asanovic, and V. Stojanovic. Silicon-photonic cros networks for global on-chip communication. In *2009 3rd ACM/IEEE International Symposium on Networks-on-Chip*, pages 124–133, 2009.
- [81] N. P. Jouppi et al. In-datacenter performance analysis of a tensor processing unit. 2017.

- [82] F. N. U. Juanda et al. A 10-gs/s 4-bit single-core digital-to-analog converter for cognitive ultrawidebands. *TCS*, 2017.
- [83] H. Kantz and T. Schreiber. *Nonlinear Time Series Analysis*. Cambridge University Press, 2 edition, 2003.
- [84] Y.-H. Kao and H. J. Chao. Design of a bufferless photonic cros network-on-chip architecture. *IEEE Transactions on Computers*, 63(3):764–776, 2014.
- [85] V. S. P. Karempudi et al. Design exploration and scalability analysis of a cmos-integrated, polymorphic, nanophotonic arithmetic-logic unit. In *CENSS*, 2021.
- [86] V. S. P. Karempudi et al. Photonic networks-on-chip employing multilevel signaling: A cross-layer comparative study. *JETC*, 2022.
- [87] 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, GLSVLSI '20*, page 247–252, New York, NY, USA, 2020. Association for Computing Machinery.
- [88] V. S. P. Karempudi, F. Sunny, I. G. Thakkar, S. V. R. Chittamuru, M. Nikdast, and S. Pasricha. Photonic networks-on-chip employing multilevel signaling: A cross-layer comparative study. *J. Emerg. Technol. Comput. Syst.*, 18(3), mar 2022.
- [89] A. Katumba, M. Freiburger, P. Bienstman, and J. Dambre. A multiple-input strategy to efficient integrated photonic reservoir computing. *Cognitive Computation*, 9(3):307–314, Apr. 2017.
- [90] H.-U. Kim and J.-K. Kang. High-speed serial interface using pwam signaling scheme. In *2022 19th International SoC Design Conference (ISOCC)*, pages 255–256. IEEE, 2022.
- [91] R. Krishnamoorthi. Quantizing deep convolutional networks for efficient inference: A whitepaper. *arXiv preprint arXiv:1806.08342*, 2018.
- [92] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc., 2012.
- [93] H. Kwon et al. Maeri: Enabling flexible dataflow mapping over dnn accelerators via reconfigurable interconnects. In *ASPLOS*, 2017.
- [94] S. Kyle et al. Albireo: Energy-efficient acceleration of convolutional neural networks via silicon photonics. In *ISCA*, 2021.

- [95] M. Lan, M. Li, J. Xiong, W. Liu, C. Liu, and K. Li. Automated optical accelerator search: Expediting green and ubiquitous dnn-powered intelligence. *IEEE Design and Test*, 40(6):175–184, 2023.
- [96] Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *Nature*, 521(7553):436–444, May 2015.
- [97] B. G. Lee et al. All-optical comb switch for multiwavelength message routing in silicon photonic networks. *IEEE Photonics Technology Letters*, 2008.
- [98] J. Lee, C. Killian, S. L. Beux, and D. Chillet. Approximate nanophotonic interconnects. In *Proceedings of the 13th IEEE/ACM International Symposium on Networks-on-Chip*. ACM, Oct. 2019.
- [99] J. Li, K. Bai, L. Liu, and Y. Yi. A deep learning based approach for analog hardware implementation of delayed feedback reservoir computing system. In *2018 19th International Symposium on Quality Electronic Design (ISQED)*, pages 308–313, 2018.
- [100] J. Li et al. Towards acceleration of deep convolutional neural networks using stochastic computing. In *ASP-DAC*, 2017.
- [101] P. Li et al. Computation on stochastic bit streams digital image processing case studies. *TVLSI*, 2014.
- [102] Q. Li, N. Ophir, L. Xu, K. Padmaraju, L. Chen, M. Lipson, and K. Bergman. Experimental characterization of the optical-power upper bound in a silicon microring modulator. In *2012 Optical Interconnects Conference*, pages 38–39, 2012.
- [103] S. Li, S. Dev, S. Ohlendorf, K. Jamshidi, and S. Pachnicke. Photonic reservoir computing enabled by silicon micro-rings. In *Asia Communications and Photonics Conference (ACPC) 2019*, page M4C.4. Optical Society of America, 2019.
- [104] Z. Li et al. Dscnn: Hardware-oriented optimization for stochastic computing based deep convolutional neural networks. In *ICCD*, 2016.
- [105] Z. Li, F. Liu, W. Yang, S. Peng, and J. Zhou. A survey of convolutional neural networks: Analysis, applications, and prospects. *IEEE Transactions on Neural Networks and Learning Systems*, 33(12):6999–7019, 2022.
- [106] T. Liang et al. Pruning and quantization for deep neural network acceleration: A survey. *Neurocomputing*, 2021.
- [107] X. Lin and Others. All-optical machine learning using diffractive deep neural networks. *Science*, 2018.
- [108] J. Lira et al. Implementing a hybrid sram / edram nuca architecture. In *ICHPC*, 2011.

- [109] 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)*, pages 1483–1488, 2019.
- [110] P. Luciano, C.-L. Sotiropoulou, S. Gkaitatzis, M. Viti, S. Citraro, A. Retico, P. Giannetti, and M. Dell’Orso. A hardware implementation of a brain inspired filter for image processing. *IEEE Transactions on Nuclear Science*, 64(6):1374–1381, 2017.
- [111] P. Y. Ma et al. Photonic independent component analysis using an on-chip microring weight bank. *Optics Express*, 2020.
- [112] W. Maass, T. Natschläger, and H. Markram. Real-time computing without stable states: A new framework for neural computation based on perturbations. *Neural Computation*, 14(11):2531–2560, Nov. 2002.
- [113] A. Melloni, A. Canciamilla, C. Ferrari, F. Morichetti, L. O’Faolain, T. F. Krauss, R. De La Rue, A. Samarelli, and M. Sorel. Tunable delay lines in silicon photonics: Coupled resonators and photonic crystals, a comparison. *IEEE Photonics Journal*, 2(2):181–194, 2010.
- [114] P. A. Merolla et al. A million spiking-neuron integrated circuit with a scalable communication network and interface. *Science*, 2014.
- [115] C. Mesaritakis, V. Papataxiarhis, and D. Syvridis. Micro ring resonators as building blocks for an all-optical high-speed reservoir-computing bit-pattern-recognition system. *J. Opt. Soc. Am. B*, 30(11):3048–3055, Nov 2013.
- [116] D. A. B. Miller. Meshing optics with applications. *Nature Photonics*, 11(7):403–404, June 2017.
- [117] M. Miscuglio et al. Photonic tensor cores for machine learning. *Applied Physics Reviews*, sep 2020.
- [118] V. Mnih et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, Feb 2015.
- [119] G. Mourgias-Alexandris, A. Totović, A. Tsakyridis, N. Passalis, K. Vyrsoinos, A. Tefas, and N. Pleros. Neuromorphic photonics with coherent linear neurons using dual-iq modulation cells. *Journal of Lightwave Technology*, 38(4):811–819, 2020.
- [120] F. Muñoz Martínez et al. Stif: A spatio-temporal integrated folding tree for efficient reductions in flexible dnn accelerators. *JETC*, 2023.
- [121] B. Neel, M. Kennedy, and A. Kodi. Dynamic power reduction techniques in on-chip photonic interconnects. In *Proceedings of the 25th Edition on Great Lakes Symposium on VLSI, GLSVLSI ’15*, page 249–252, New York, NY, USA, 2015. Association for Computing Machinery.

- [122] D.-R. Oh et al. An 8b 1gs/s 2.55mw sar-flash adc with complementary dynamic amplifiers. In *IVLSIC*, 2020.
- [123] K. Padmaraju et al. Intermodulation crosstalk characteristics of wdm silicon microring modulators. *IEEE Photonics Technology Letters*, 2014.
- [124] Y. Pan, J. Kim, and G. Memik. Flexishare: Channel sharing for an energy-efficient nanophotonic crossbar. In *HPCA - 16 2010 The Sixteenth International Symposium on High-Performance Computer Architecture*, pages 1–12, 2010.
- [125] Y. Paquot, F. Duport, A. Smerieri, J. Dambre, B. Schrauwen, M. Haelterman, and S. Massar. Optoelectronic reservoir computing. *Scientific Reports*, 2(1), Feb. 2012.
- [126] A. Paszke et al. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*. 2019.
- [127] N. Peserico et al. Integrated photonic tensor processing unit for a matrix multiply: a review. *JLT*, 2023.
- [128] M. Popel et al. Transforming machine translation: a deep learning system reaches news translation quality comparable to human professionals. *Nature Communications*, 11(1):4381, Sep 2020.
- [129] P. R. Prucnal et al. *Neuromorphic Photonics*. CRC Press, May 2017.
- [130] E. Qin, A. Samajdar, H. Kwon, V. Nadella, S. Srinivasan, D. Das, B. Kaul, and T. Krishna. Sigma: A sparse and irregular gemm accelerator with flexible interconnects for dnn training. In *2020 IEEE International Symposium on High Performance Computer Architecture (HPCA)*, pages 58–70, 2020.
- [131] C. Ramey. Silicon photonics for artificial intelligence acceleration : Hotchips 32. In *2020 IEEE Hot Chips 32 Symposium (HCS)*, pages 1–26, 2020.
- [132] M. Rastegari, V. Ordonez, J. Redmon, and A. Farhadi. Xnor-net: Imagenet classification using binary convolutional neural networks. In *European conference on computer vision*, pages 525–542. Springer, 2016.
- [133] A. Ren et al. Sc-dcnn: Highly-scalable deep convolutional neural network using stochastic computing. ASPLOS '17, New York, NY, USA, 2017. ACM.
- [134] A. Rizzo, Q. Cheng, S. Daudlin, and K. Bergman. Ultra-broadband silicon photonic interleaver for massive channel count frequency combs. In *2020 Conference on Lasers and Electro-Optics (CLEO)*, pages 1–2, 2020.
- [135] A. Rizzo et al. Massively scalable kerr comb-driven silicon photonic link. *Nature Photonics*, 2023.

- [136] A. Rizzo, Y. London, G. Kurczveil, T. Van Vaerenbergh, M. Fiorentino, A. Seyed, D. Livshits, R. G. Beausoleil, and K. Bergman. Energy efficiency analysis of frequency comb sources for silicon photonic interconnects. In *2019 IEEE Optical Interconnects Conference (OI)*, pages 1–2, 2019.
- [137] S. Samarasinghe. *Neural Networks for Applied Sciences and Engineering: From Fundamentals to Complex Pattern Recognition*. Auerbach, 2007.
- [138] M. Sandler et al. Inverted residuals and linear bottlenecks: Mobile networks for classification, detection and segmentation. *CoRR*, 2018.
- [139] J. Schmidhuber. Deep learning in neural networks: An overview. 2014.
- [140] B. Schrauwen, D. Verstraeten, and J. Campenhout. An overview of reservoir computing: Theory, applications and implementations. pages 471–482, 01 2007.
- [141] M. A. Seyed, R. Wu, C.-H. Chen, M. Fiorentino, and R. G. Beausoleil. 15 gb/s transmission with wide-fsr carrier injection ring modulator for tb/s optical links. In *Conference on Lasers and Electro-Optics*, page SF2F.7. Optical Society of America, 2016.
- [142] B. J. Shastri, A. N. Tait, T. F. de Lima, W. H. P. Pernice, H. Bhaskaran, C. D. Wright, and P. R. Prucnal. Photonics for artificial intelligence and neuromorphic computing. *Nature Photonics*, 15(2):102–114, Jan. 2021.
- [143] S. Shekhar. Silicon photonics: A brief tutorial. *IEEE Solid-State Circuits Magazine*, 13(3):22–32, 2021.
- [144] Y. Shen, N. C. Harris, S. Skirlo, M. Prabhu, T. Baehr-Jones, M. Hochberg, X. Sun, S. Zhao, H. Larochelle, D. Englund, and M. Soljačić. Deep learning with coherent nanophotonic circuits. *Nature Photonics*, 11(7):441–446, June 2017.
- [145] K. Shiflett, D. Wright, A. Karanth, and A. Louri. Pixel: Photonic neural network accelerator. In *2020 IEEE International Symposium on High Performance Computer Architecture (HPCA)*, pages 474–487, 2020.
- [146] H. Shoman, H. Jayatilleka, A. H. K. Park, N. A. F. Jaeger, S. Shekhar, and L. Chrostowski. Compact silicon microring modulator with tunable extinction ratio and wide fsr. In *Optical Fiber Communication Conference*, page Tu2E.1. Optical Society of America, 2018.
- [147] Y.-S. Shu. A 6b 3gs/s 11mw fully dynamic flash adc in 40nm cmos with reduced number of comparators. In *VLSIC*, 2012.
- [148] X. Si et al. A twin-8t sram computation-in-memory unit-macro for multibit cnn-based ai edge processors. *IJSSC*, 2020.
- [149] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

- [150] 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*, 378(6617):270–276, 2022.
- [151] M. C. Soriano, D. Brunner, M. Escalona-Morán, C. R. Mirasso, and I. Fischer. Minimal approach to neuro-inspired information processing. *Frontiers in Computational Neuroscience*, 9:68, 2015.
- [152] 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*, 41(11):4337–4348, 2022.
- [153] C. Sugano, K. Kanno, and A. Uchida. Reservoir computing using multiple lasers with feedback on a photonic integrated circuit. *IEEE Journal of Selected Topics in Quantum Electronics*, 26(1):1–9, 2020.
- [154] C. Sun et al. A 45 nm cmos-soi monolithic photonics platform with bit-statistics-based resonant microring thermal tuning. *IJSSC*, 2016.
- [155] C. Sun, M. T. Wade, Y. Lee, J. S. Orcutt, L. Alloatti, M. S. Georgas, A. S. Waterman, J. M. Shainline, R. R. Avizienis, S. Lin, B. R. Moss, R. Kumar, F. Pavanello, A. H. Atabaki, H. M. Cook, A. J. Ou, J. C. Leu, Y.-H. Chen, K. Asanović, R. J. Ram, M. A. Popović, and V. M. Stojanović. Single-chip microprocessor that communicates directly using light. *Nature*, 528(7583):534–538, Dec. 2015.
- [156] F. Sunny et al. A silicon photonic accelerator for convolutional neural networks with heterogeneous quantization. In *GLSVLSI*, 2022.
- [157] 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)*, pages 1069–1074, 2021.
- [158] F. Sunny, A. Mirza, I. Thakkar, S. Pasricha, and N. Mahdi. Lorax: Loss-aware approximations for energy-efficient silicon photonic networks-on-chip, 2020.
- [159] F. P. Sunny et al. A survey on silicon photonics for deep learning. *JETC*, 2021.
- [160] F. P. Sunny, A. Mirza, M. Nikdast, and S. Pasricha. Robin: A robust optical binary neural network accelerator. *ACM Trans. Embed. Comput. Syst.*, 20(5s), sep 2021.
- [161] V. Sze et al. Efficient processing of deep neural networks: A tutorial and survey. *Proceedings of the IEEE*, 2017.
- [162] C. Szegedy et al. Going deeper with convolutions. In *CVPR*, 2015.

- [163] A. N. Tait, T. F. de Lima, M. A. Nahmias, H. B. Miller, H.-T. Peng, B. J. Shastri, and P. R. Prucnal. A silicon photonic modulator neuron. 2018.
- [164] A. N. Tait et al. Broadcast and weight: An integrated network for scalable photonic spike processing. *J. Lightwave Technol.*, 2014.
- [165] A. N. Tait et al. Microring weight banks. *JSTQE*, 2016.
- [166] A. N. Tait et al. Neuromorphic photonic networks using silicon photonic weight banks. *Scientific reports*, 2017.
- [167] A. N. Tait, H. Jayatilleka, T. F. D. Lima, P. Y. Ma, M. A. Nahmias, B. J. Shastri, S. Shekhar, L. Chrostowski, and P. R. Prucnal. Feedback control for microring weight banks. *Opt. Express*, 26(20):26422–26443, Oct 2018.
- [168] A. N. Tait, M. A. Nahmias, B. J. Shastri, M. P. Chang, A. X. Wu, E. Zhou, E. C. Blow, T. F. de Lima, B. Wu, and P. R. Prucnal. Balanced wdm weight banks for analog optical processing and networking in silicon. In *2015 IEEE Summer Topicals Meeting Series (SUM)*, pages 110–111, 2015.
- [169] A. N. Tait, M. A. Nahmias, B. J. Shastri, and P. R. Prucnal. Broadcast and weight: An integrated network for scalable photonic spike processing. *J. Lightwave Technol.*, 32(21):3427–3439, Nov 2014.
- [170] M. Tan et al. Efficientnet: Rethinking model scaling for convolutional neural networks. *CoRR*, 2019.
- [171] A. D. Team. Hub: A dataset format for ai. a simple api for creating, storing, collaborating on ai datasets of any size and streaming them to ml frameworks at scale. *GitHub*. Note: <https://github.com/activeLOOPai/Hub>, 2022.
- [172] I. Thakkar, S. Vineel, and S. Pasricha. Run-time laser power management in photonic nocs with on-chip semiconductor optical amplifiers. pages 1–4, 09 2016.
- [173] I. G. Thakkar, S. V. R. Chittamuru, and S. Pasricha. Mitigation of homodyne crosstalk noise in silicon photonic noc architectures with tunable decoupling. In *2016 International Conference on Hardware/Software Codesign and System Synthesis (CODES+ISSS)*, pages 1–10, 2016.
- [174] I. G. Thakkar, S. V. R. Chittamuru, and S. Pasricha. Improving the reliability and energy-efficiency of high-bandwidth photonic noc architectures with multilevel signaling. In *2017 Eleventh IEEE/ACM International Symposium on Networks-on-Chip (NOCS)*, pages 1–8, 2017.
- [175] I. G. Thakkar, S. Vineel Reddy Chittamuru, and S. Pasricha. A comparative analysis of front-end and back-end compatible silicon photonic on-chip interconnects. In *2016 ACM/IEEE International Workshop on System Level Interconnect Prediction (SLIP)*, pages 1–8, 2016.

- [176] T. N. Theis and H.-S. P. Wong. The end of moore’s law: A new beginning for information technology. *Computing in Science Engineering*, 19(2):41–50, 2017.
- [177] C. A. Thraskias, E. N. Lallas, N. Neumann, L. Schares, B. J. Offrein, R. Henker, D. Plettemeier, F. Ellinger, J. Leuthold, and I. Tomkos. Survey of photonic and plasmonic interconnect technologies for intra-datacenter and high-performance computing communications. *IEEE Communications Surveys Tutorials*, 20(4):2758–2783, 2018.
- [178] J. Torrejon, M. Riou, F. A. Araujo, S. Tsunegi, G. Khalsa, D. Querlioz, P. Bortolotti, V. Cros, K. Yakushiji, A. Fukushima, H. Kubota, S. Yuasa, M. D. Stiles, and J. Grollier. Neuromorphic computing with nanoscale spintronic oscillators. *Nature*, 547(7664):428–431, July 2017.
- [179] A. R. Totović, G. Dabos, N. Passalis, A. Tefas, and N. Pleros. Femtojoule per mac neuromorphic photonics: An energy and technology roadmap. *IEEE Journal of Selected Topics in Quantum Electronics*, 26(5):1–15, 2020.
- [180] G. Urbain, J. Degrave, B. Carette, J. Dambre, and F. Wyffels. Morphological properties of mass–spring networks for optimal locomotion learning. *Frontiers in Neurorobotics*, 11:16, 2017.
- [181] A. van den Bosch et al. A 10-bit 1-gsample/s nyquist current-steering cmos d/a converter. *JSSC*, 2001.
- [182] S. Van Winkle, A. K. Kodi, R. Bunescu, and A. Louri. Extending the power-efficiency and performance of photonic interconnects for heterogeneous multi-cores with machine learning. In *2018 IEEE International Symposium on High Performance Computer Architecture (HPCA)*, pages 480–491, 2018.
- [183] K. Vandoorne, J. Dambre, D. Verstraeten, B. Schrauwen, and P. Bienstman. Parallel reservoir computing using optical amplifiers. *IEEE Transactions on Neural Networks*, 22(9):1469–1481, 2011.
- [184] K. Vandoorne, P. Mechet, T. V. Vaerenbergh, M. Fiers, G. Morthier, D. Verstraeten, B. Schrauwen, J. Dambre, and P. Bienstman. Experimental demonstration of reservoir computing on a silicon photonics chip. *Nature Communications*, 5(1), Mar. 2014.
- [185] J. Vatin, D. Rontani, and M. Sciamanna. High-speed all-optical reservoir computing using vcsel polarization dynamics. In *2018 European Conference on Optical Communication (ECOC)*, pages 1–3, 2018.
- [186] S. Vatsavai et al. Sconna: A stochastic computing based optical accelerator for ultra-fast, energy-efficient inference of integer-quantized cnns. In *IPDPS*, 2023.
- [187] S. S. Vatsavai et al. Silicon photonic microring based chip-scale accelerator for delayed feedback reservoir computing. In *VLSID*, 2021.

- [188] S. S. Vatsavai et al. An optical xnor-bitcount based accelerator for efficient inference of binary neural networks. In *ISQED*, pages 1–8. IEEE, 2023.
- [189] 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)*, pages 1–8. IEEE, 2020.
- [190] S. S. Vatsavai and I. Thakkar. A bit-parallel deterministic stochastic multiplier. In *ISQED*, 2023.
- [191] D. Verstraeten, B. Schrauwen, M. D’Haene, and D. Stroobandt. An experimental unification of reservoir computing methods. *Neural Networks*, 20(3):391–403, 2007. Echo State Networks and Liquid State Machines.
- [192] C.-C. Wang et al. 67.5-fj per access 1-kb sram using 40-nm logic cmos process. In *ISCAS*, 2021.
- [193] S. Wang, A. Novick, A. Rizzo, R. Parsons, S. Sanyal, K. J. McNulty, B. Y. Kim, Y. Okawachi, Y. Wang, A. Gaeta, M. Lipson, and K. Bergman. Integrated, compact, and tunable band-interleaving of a kerr comb source. In *CLEO 2023*, page STh3J.6. Optica Publishing Group, 2023.
- [194] Z. Wang, H. Gu, Y. Yang, and B. Zhang. Power allocation method for tdm-based optical network on chip. *IEEE Photonics Technology Letters*, 25(10):973–976, 2013.
- [195] B. Wu et al. Mixed precision quantization of convnets via differentiable neural architecture search. *arXiv preprint arXiv:1812.00090*, 2018.
- [196] D. Wu et al. Ugemm: Unary computing architecture for gemm applications. In *ISCA*, 2020.
- [197] R. Wu, C.-H. Chen, J.-M. Fedeli, M. Fournier, R. G. Beausoleil, and K.-T. Cheng. Compact modeling and system implications of microring modulators in nanophotonic interconnects. In *2015 ACM/IEEE International Workshop on System Level Interconnect Prediction (SLIP)*, pages 1–6, 2015.
- [198] X. Xu, Y. Ding, S. X. Hu, M. Niemier, J. Cong, Y. Hu, and Y. Shi. Scaling for edge inference of deep neural networks. *Nature Electronics*, 1(4):216–222, Apr. 2018.
- [199] X. Xu et al. 11 TOPS photonic convolutional accelerator for optical neural networks. *Nature*, Jan. 2021.
- [200] C.-Y. Yang and Y. Lee. A pwm and pam signaling hybrid technology for serial-link transceivers. *IEEE Transactions on Instrumentation and Measurement*, 57(5):1058–1070, 2008.

- [201] L. Yang, L. Zhang, and R. Ji. On-chip optical matrix-vector multiplier. *Proc SPIE*, 8855, 09 2013.
- [202] Yann et al. Deep learning. *Nature*, May 2015.
- [203] H. Ye et al. Double-gate w-doped amorphous indium oxide transistors for monolithic 3d capacitorless gain cell edram. In *IEDM*, 2020.
- [204] Y. Yu et al. Opu: An fpga-based overlay processor for convolutional neural networks. *IVLSI Systems*, 2020.
- [205] 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*, page 373–390, Berlin, Heidelberg, 2018. Springer-Verlag.
- [206] H. Zhang et al. An optical neural chip for implementing complex-valued neural network. *Nature Communications*, 2021.
- [207] W. Zhang, C. Huang, H.-T. Peng, S. Bilodeau, A. Jha, E. Blow, T. F. de Lima, B. J. Shastri, and P. Prucnal. Silicon microring synapses enable photonic deep learning beyond 9-bit precision. *Optica*, 9(5):579–584, May 2022.
- [208] X. Zhang et al. Shufflenet: An extremely efficient convolutional neural network for mobile devices. *CoRR*, 2017.
- [209] X. Zhang et al. Architecting a stochastic computing unit with molecular optical devices. In *ISCA*, 2018.
- [210] X. Zhao et al. An integrated optical neural network chip based on mach-zehnder interferometers. 2018.
- [211] Z. Zhao, D. Liu, M. Li, Z. Ying, L. Zhang, B. Xu, B. Yu, R. T. Chen, and D. Z. Pan. Hardware-software co-design of slimmed optical neural networks. In *Proceedings of the 24th Asia and South Pacific Design Automation Conference, ASPDAC '19*, page 705–710, New York, NY, USA, 2019. Association for Computing Machinery.
- [212] L. Zhou and A. K. Kodi. Probe: Prediction-based optical bandwidth scaling for energy-efficient nocs. In *2013 Seventh IEEE/ACM International Symposium on Networks-on-Chip (NoCS)*, pages 1–8, 2013.
- [213] T. Zhou et al. Large-scale neuromorphic optoelectronic computing with a re-configurable diffractive processing unit. *Nature Photonics*, 2021.
- [214] 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)*, pages 1966–1976, 2020.

- [215] 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)*, pages 1438–1443, 2020.
- [216] B. Zoph et al. Learning transferable architectures for scalable image recognition. *CoRR*, 2017.

Vita

Sairam Sri Vatsavai

Education

- University of Kentucky, Lexington, USA
PhD in Electrical Engineering, Expected May 2024.
- Jawaharlal Nehru Technological University, Hyderabad, India
Bachelor of Technology, Electronics and Communication, May 2016.

Provisional Patent

- ***STOCHASTIC COMPUTING ENABLED OPTICAL HARDWARE ARCHITECTURES FOR ENERGY EFFICIENT AND SCALABLE ACCELERATION OF DEEP NEURAL NETWORKS***, Ishan Thakkar, Sairam Sri Vatsavai and Venkata Sai Praneeth Karempudi, Provisional Patent, University of Kentucky (Application # 63/560,313), November, 2023.

Publications

Journal Publications

- Sairam Sri Vatsavai, Venkata Sai Praneeth Karempudi, Ishan Thakkar, ***HEANA: A Hybrid Time-Amplitude Analog Optical Accelerator with Flexible Dataflows for Energy-Efficient CNN Inference*** ACM TODAES, Submitted on: November, 2023. (Under Revision).
- Sairam Sri Vatsavai and I. G. Thakkar, ***Photonic Reconfigurable Accelerators for Efficient Inference of CNNs With Mixed-Sized Tensors***, in IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 41, no. 11, pp. 4337-4348, Nov. 2022, doi: 10.1109/TCAD.2022.3197538.

Conference Publications

- Sairam Sri Vatsavai, Venkata Sai Praneeth Karempudi, and Ishan Thakkar. ***An Electro-Photonic Unary Multiply-Accumulate (MAC) Circuit***, Accepted at Conference on Lasers and Electro-Optics (CLEO) 2024.
- Sairam Sri Vatsavai, Venkata Sai Praneeth Karempudi, and Ishan Thakkar. ***A Comparative Analysis of Microring Based Incoherent Photonic GEMM Accelerators***, Accepted at IEEE ISQED 2024.

- Sairam Sri Vatsavai, V. S. P. Karempudi, I. Thakkar, A. Salehi and T. Hastings, ***SCONNA: A Stochastic Computing Based Optical Accelerator for Ultra-Fast, Energy-Efficient Inference of Integer-Quantized CNNs***, 2023 IEEE International Parallel and Distributed Processing Symposium (IPDPS), St. Petersburg, FL, USA, 2023, pp. 546-556, doi: 10.1109/IPDPS54959.2023.00061.
- Sairam Sri Vatsavai, V. Karempudi, I. Thakkar, A. Salehi and T. Hastings, ***SCONNA: A Stochastic Computing Based Optical Accelerator for Ultra-Fast, Energy-Efficient Inference of Integer-Quantized CNNs***, in 2023 IEEE International Parallel and Distributed Processing Symposium (IPDPS), St. Petersburg, FL, USA, 2023 pp. 546-556.
- Sairam Sri Vatsavai, V. Sai Praneeth Karempudi and I. Thakkar, ***An Optical XNOR-Bitcount Based Accelerator for Efficient Inference of Binary Neural Networks***, 2023 24th International Symposium on Quality Electronic Design (ISQED), San Francisco, CA, USA, 2023, pp. 1-8, doi: 10.1109/ISQED57927.2023.10129294.
- Sairam Sri Vatsavai and I. Thakkar, ***A Bit-Parallel Deterministic Stochastic Multiplier***, 2023 24th International Symposium on Quality Electronic Design (ISQED), San Francisco, CA, USA, 2023, pp. 1-1, doi: 10.1109/ISQED57927.2023.10129297.
- Sairam Sri Vatsavai and I. Thakkar, ***Silicon Photonic Microring Based Chip-Scale Accelerator for Delayed Feedback Reservoir Computing***, 2021 34th International Conference on VLSI Design and 2021 20th International Conference on Embedded Systems (VLSID), Guwahati, India, 2021, pp. 129-134, doi: 10.1109/VLSID51830.2021.00027.
- Sairam Sri 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***, 2020 14th IEEE/ACM International Symposium on Networks-on-Chip (NOCS), Hamburg, Germany, 2020, pp. 1-8, doi: 10.1109/NOCS50636.2020.9241712.