Performance and Energy Evaluation of Parallelization Strategies for Network on Chip Communication Architectures: Case Study of Canny Edge Detector Application

By

Harikrishna Menon Thelakkat

A Thesis

In

The Department

Of

Electrical and Computer Engineering

Presented in Partial Fulfillment of the Requirements for the

Degree of Master of Applied Science (Electrical and Computer Engineering)

at

Concordia University

Montréal, Québec, Canada

June 2018

CONCORDIA UNIVERSITY
SCHOOL OF GRADUATE STUDIES

This is to certify that the thesis prepared

By: Harikrishna Menon Thelakkat

Entitled:

"Performance and Energy Evaluation of Parallelization Strategies for Network on Chip Communication Architectures: Case Study of Canny Edge Detector Application"

and submitted in partial fulfillment of the requirements for the degree of

Master of Applied Science (Electrical and Computer Engineering)

Complies with the regulations of this University and meets the accepted standards with respect to originality and quality.

Signed by the final examining committee:

_____ Chair
Dr. D. Qiu

_____ Examiner, External
Dr. L Narayanan to the Program

_____ Examiner
Dr. Y. Liu

_____ Supervisor
Dr. Jelena Trajkovic

Approved by: _____
Dr. W. E. Lynch, Chair
Department of Electrical and Computer Engineering

June 2018

_____
Dr. Amir Asif
Dean, Faculty of Engineering and Computer Science

ABSTRACT

The substantial amount of research pertaining to the usage of optical networks for communication between cores in a multicore processor underlines the need for effective communication schemes. This necessitates the exploration of the efficiency of an optical network with a suitable benchmark which compares the different features essential for having an effective communication between the cores. As far as communication in a network is considered, the parameters that are most crucial are the delays and energy consumption. This thesis focuses on an industrial-sized application from the image processing field, Canny Edge Detector, to compare the performance in terms of network parameters which are the contention delay, latency and the energy consumption with the different settings on the network on chip simulator. The Canny Edge Detector application is implemented with various software parallelization schemes for better performance as compared to the normal serialized application. Also, to analyze the effectiveness of multicore processors, a comparison among sequential and parallelized coding techniques is performed in this thesis.

Software parallelization schemes applied to the algorithm executed on optical network architectures improve the latency and delay of the network up to 60% in the best case, while the total energy consumption values have a worst case overhead of around 50%. For almost all the configuration parameters, the parallelized schemes provide much better results for the outputs than the sequential implementation. The design parameters help determine the optimal amount of resources required for efficient execution of an image processing algorithm using a moderate to heavy workload on an NoC based on the minimal delay and energy consumption values.

ACKNOWLEDGEMENTS

First of all, I would like to express my sincere gratitude towards my Supervisor, Dr. Jelena Trajkovic, for providing me an opportunity to work under her on this thesis. I also thank her for her constant support and guidance without which this would never have been possible. My work under her has improved my technical knowledge and presentation skills, while helping me embark on my desired career path for which I am ever so grateful.

I would like to thank my examining committee members, Dr. Lata Narayanan and Dr. Yan Liu for reviewing my thesis and for their invaluable suggestions. I am also grateful to all the professors who taught me at Concordia University. They have helped me strengthen my knowledge on the fundamental concepts and introduced me to new ideas and challenges.

I would like to extend my thanks to the CSAIL group form MIT for the provision of Graphite multi-core network on chip simulator, which has formed an integral part of all the experiments in this thesis.

Next, I would like to express my gratitude towards all my lab mates, for this whole period of time for all their aid and assistance. I am especially thankful to Sara Karimi, for her unmatched support and help. My friends – Paul Leons, Alex Joseph and Sarath Somasekharan have been a great help and guidance from the beginning of my life in Canada and have provided immense support throughout my master's program for which I am much indebted to them. I am extremely thankful to all my family members, especially my mom, dad and grandmother who have always believed in me and provided me with all the moral support. I dedicate this thesis work to all three of them.

Last but not least, I would like to thank God Almighty for giving the strength and knowledge to pursue this endeavor and successfully complete it.

CONTENTS

List of Equations

List of Tables

List of Figures

Chapter 1

Introduction

The advent of multicore processors has enforced two significant changes in the perception of solving complex computational challenges. Primarily, applications need to be developed keeping in mind the extent of software parallelization that can be extracted from each processing element. The main emphasis from the software perspective is modularization of the task at hand to be processed by multiple processing elements. Parallel computing platforms such as Nvidia CUDA[1] and Intel TBB[2] provide an environment for development of parallelizable code for multiple core graphics processing units and multicore processors respectively for the existing hardware.

The second area that demands attention is the requirement of novel and effective communication strategies for the effective distribution of tasks to the different cores to effectively address resource requirements. With the ever increasing numbers of processing cores that can be accommodated on a single chip, the communication among the processing units remains one of the main bottlenecks affecting the performance. Research on this field has unveiled the idea of networks on chip which has introduced different network architectures for managing the network traffic. Network on chip architectures have improved the scalability and performance of multicore systems. The observations from [3] indicate the advantages offered by photonic networks on chip, which have been enabled by advancements in the field of silicon photonics. They claim to have lesser power requirements and can provide higher bandwidth for communication of data between the cores. The field of optical networks on chip, which is relatively newer as compared to

electrical NoCs and still under extensive research, require benchmarks and applications to validate their efficiency in different areas for software simulations.

An effective implementation of software parallelized application can improve the performance and reduce the latency[4]. It can be used to compare the performance of different network on chip architectures and also enables the hardware designers to choose the optimal design parameters.

1.1 Motivation and Problem Statement

The field of optical network on chips offers vast avenues for exploration. Extensive research on the development of viable photonic network on chip architectures with varying routing strategies and communication protocols has been continuing over the past few years. Before the development of chips integrated with photonic networks, the efficiency of different architectures needs to be studied to tailor the needs of hardware designers.

Although there are other benchmarks that support multithreaded workloads, such as SPLASH [5] and PARSEC [6] do not have multiple parallelized versions of the same application. Having different parallelized versions of the application provides the freedom for software designers to choose the type of implementation based on their requirements. Software simulations can estimate the behaviour of an application on the actual hardware. Moreover they also provide the flexibility of running multiple test cases for multiple configurations and provide valuable data which forms the base for actual implementation in hardware. Hardware designers can use the simulation results from a benchmark to choose the NoC architectures to best suit their requirements. Also, comparison of NoC architectures in terms of software parallelized versions of a single application has not been performed yet.

The impact of different configuration parameters on the overall efficiency also needs to be studied to extract maximum hardware performance.

Hardware designers mostly have specific design requirements pertaining to resource utilization and trade-off between performance parameters like latency and power consumption for their application. Even though generalized experiments can provide an approximate estimate on the performance of a particular application in hardware, the demand for more accurate results necessitate the exploration of that specific application and generation of results based on them. The integration of photonic components on the silicon based chips can enable bring the efficiency of optics on intra chip communication.

The advancements in the field of technology and medicine have enabled the use of image processing applications for various needs and one such application uses the aid of edge detection in laproscopic surgery that requires very good accuracy and throughput[4]. Performing a surgery requires high precision and accuracy as mistakes would be life threatening. Since the surgical procedures need to be performed in real time, the communication needs to be efficient and real time. Canny edge detection is chosen in thesis keeping in mind the relative importance and for this particular usage in the medical field.

1.2 Proposed Solution

In order to extract the maximum performance and optimal resource utilization from the network architectures, it is necessary to have an application that effectively incorporates the different software parallelization schemes. Hence, an edge detection application, one of the most common and significant image processing algorithms, is chosen as the application to perform the experiments for its use in the field of laproscopic surgery which requires fast

processing. Moreover, the application has multiple stages of implementation and has the scope to allow different types of software parallelization techniques to extract maximum performance. This caters to the need of fully utilizing the parallel programming capability of a multicore chip.

The performance results are extracted based on the software implementation as well as the variation of the NoC design parameters. It is expected to provide invaluable information to the hardware designers regarding the choice of NoC architecture and the resources to be considered based on their design constraints.

1.3 Thesis Contribution

This thesis provides the following contributions:

- Implementing different software parallelization schemes to the Canny edge detection algorithm to estimate the optimal resource utilization.
- Comparison of chosen NoC and ONoC architectures to quantify the performance metrics like delay and energy consumption for the Canny edge detection algorithm.
- Provision of the software parallelized versions of the Canny edge detector application in an open source manner for benchmarking.

As mentioned above, this work aims at employing the Canny edge detection application, one of the simplest and most accurate edge detection application for the performance comparison of different NoC architectures, predominantly classified as electrical and optical. Our experiments address the need to explore an application which is optimized in terms of the parallelization schemes used for software development for observing the performance of network on chip architectures. Experiments are performed on the developed

application to compare and contrast the network features that influence the execution of an application on a multicore platform. Apart from helping to analyze the effectiveness of software parallelization of the algorithm in comparison with the sequential implementation, it also acts as an effective benchmark for the comparison of different on chip architectures. This also helps the designers choose the desired configuration for their architecture and the nature of implementation. It also explores the performance of network architectures based on variations to many input design parameters. Moreover, we hope that both analysis of the different parallel implantations of the Canny edge detection application and the analysis of different network-on-chip and communication parameters could be useful for the future designers in order to facilitate shorter time-to-market.

1.4 Outline

Our primary goal was to choose an application with widespread usage and that offers sufficient room to employ different parallelization strategies. Edge detection applications form the preliminary stages for many image processing, computer vision and machine learning applications for image segmentation and extraction of data[7]. We chose the Canny edge detection [8] algorithm to implement as our application because of its high accuracy and the parallelization schemes that can be applied to the different stages of the algorithm. The first step in the implementation involves development of the application and its execution in the native system. This process is carried out for all the three different implementations and observation of results.

The second stage involved porting of the algorithm to a multicore Network on Chip (NoC) simulator to analyze and compare the chosen outputs for the NoCs. We use the Graphite

multicore simulator[9], which extracts the network performance based on different configuration parameters.

1.5 Related Work

This thesis explores related work in 4 different domains, which are: the necessity for Networks on Chip, Optical Networks on Chip and their advantages over NoC, comparison of NoC simulators and the application used for implementation including their parallelization strategies.

1.5.1 Networks on Chip

The trend of using multi-core processors at a lesser clock frequency to increase the chip performance has introduced packet-based routing networks for global intra-chip communication, commonly known as network on chip. The increase in the number of processing elements that can be accommodated on a chip has increased the shared buses can be easily implemented, has a simple topology and low area cost. The different types of communication networks on a chip include shared buses, direct networks, indirect networks and hybrid networks[10]. The direct networks have direct connection between each of the cores which does not require any specific routing mechanism, while the indirect networks operate with routing schemes among nodes to provide full connectivity. The hybrid networks utilize a mixture of both the direct and indirect networks. The increased contention delays, the energy consumption, larger latency and lower bandwidth proves to be disadvantageous for a larger sized network in the case of shared buses [11]. The crossbar networks[10], a form of indirect network that has indirect connection between all the cores were introduced as an improvement over the shared buses, which provide better bandwidth and reduces the contention for data path. But, the advantages of shared buses are not reciprocated by

crossbars as they have larger energy consumption, larger global wire placement issues in case of a large network and have lesser scalability[10]. With further increase in the size of on-chip networks, the requirement for increased bandwidths, lesser contention and data latency, lesser global wires to reduce area consumption needed to be introduced. The 2D mesh topology provides direct connection between the cores and is one of the easiest yet efficient implementation of the above requirements at the expense of routers for proper routing between cores[10] .

1.5.2 Optical Networks on Chip Architectures

The electrical NoCs restrict the scalability due to increase in the propagation delay and losses due to inductive and capacitive coupling[12] and limited bandwidth. This has inspired researchers to work with opto-electrical components to create photonic network on chip and its advantages techniques over the conventional electrical networks have been a subject of research for many publications. The idea and improvements offered by photonic networks on chip over the electrical networks are given in [13]. The challenge of scaling the bandwidth and minimizing latency keeping the power consumption values in check is identified in [3] and a hybrid photonic network on chip with electrical network for the local communication while utilizing the optical network for global communication. A 2x2 photonic switching matrix that uses micro ring resonators is used for the photonic network implementation is presented in [3]. There are many optical network on chip architectures proposed like Corona[14], CHAMELEON (CHANNEL Efficient Optical Network On Chip)[15], Firefly[16], ATAC[17] and ORNOC[12] each with their unique style of integration of photonic and electrical components for creating a communication network within the chip. While [18] proposes the usage of an electrical control network and photonic data transmission which causes contention in the system, [19], [20] proposes architectures that do

not allow wavelength reuse and hence, affects the scalability. Corona[14] performs multiple writes and hence, needs arbitration to solve for write conflicts in the network. Firefly[16] tries to reduce the power consumption by using a separate data initialization packet for switching on the data receiver resources. This causes additional latency and in turn rapidly reduces throughput. FlexiShare[21] uses token stream based arbitration scheme to transfer the packets. CHAMELEON[22] provides an extension to the ORNoC architecture by opening point-to-point connections at run time, using a reconfiguration layer thereby increasing the bandwidth with respect to the network traffic and reducing the power consumption in the optical network. But, it requires a separate electrical layer to act as a control network over the optical network for the configuration process. This causes additional area and power overhead, hence making it disadvantageous. ATAC[17] uses a cluster (refers to a group of nodes) based architecture with all the nodes. ORNoC, similar to ATAC uses a cluster based arrangement, but uses a ring based routing scheme which assigns specific wavelengths for communication between the cores while allowing reuse of wavelengths in possible cases, to reduce power consumption. Both ATAC and ORNoC propose contention-free architecture, where the data packets do not need arbitration before they are transferred via the network. Out of these, ATAC and ORNoC are considered for evaluation in this thesis are as they have arbitration free architecture and do not require a separate control layer and hence, reduces power and area consumption.

1.5.3 NoC Simulators

The research in the area of photonics NoCs is still progressing and the performance evaluation makes use of different multi core simulators. NocSim[23] implements different electrical network topologies like the torus and star topologies exclusively for electrical NoCs. Orion[24] implements the power and area modelling for electrical networks. It has a

lot of shortcomings like incomplete architecture and router timings, not being scalable, very low accuracy for current models and incapability in the expansion of current models. PhoenixSim[25] introduces a parametrized modeling of optical networks. But, it uses the electrical modelling from Orion, which includes all its disadvantages, and moreover does not have suitable interconnection between electrical and photonic networks. Graphite[9] is a comprehensive multicore simulator which implements both electrical and photonic NoCs. It uses Design Space Exploration for Networks Tool (DSENT) [26] for evaluating the power and delay consumption in the network. DSENT provides a good interface between optical and electrical networks and also models the networks better as compared to ORION. Graphite, irrespective of not being a cycle accurate simulator, implements synchronization schemes to give almost near cycle accurate performance with lesser simulation time as compared to the cycle accurate simulators. The three synchronization schemes supported by Graphite are: the lax, the lax with barrier synchronization and the lax with point-to-point synchronization. While the lax synchronization assigns local clocks to each of the cores, the lax barrier scheme waits on a fixed number of cycles to synchronize the execution of threads. In the lax with point-to-point synchronization, each tile periodically synchronizes with another tile and when the difference is greater than a certain number of cycles, the tile that is ahead sleeps for some time. Graphite achieves 41x slowdown compared to execution in the native machine while the cycle accurate simulators require a 1000x to 100,000x slowdown overhead for increased accuracy. Hence, we use the Graphite simulator for analysis of the results and comparison of the NoC architectures.

1.5.4 Application and Parallelization strategies

The Canny edge detection[8] application has been implemented in CUDA[27] and used for evaluating improvement in the performance of GPUs over CPUs in[28]. The application is

also used to study its performance on multicore central processing units and many core graphics processing units in [9], which implements the different steps for edge detection with different parallelization schemes. This approach, along with sequential[29] and pure data parallelism approaches has been implemented to observe and evaluate their performance on the network on chip architectures. While SPLASH benchmark suite has more applications mainly in the field of High Performance Computing like *barnes, cholesky, fmm, lu* and *ocean*, PARSEC benchmark suite supports applications in more variety of fields like Computer Vision, Data Mining, Media Processing and Financial Analysis. While the SPLASH[5] and PARSEC[6] benchmarks are provided in the simulator for performance comparison of the architectures, they do not implement different parallelization schemes for the same application. We compare three different types of software parallelized implementations of the Canny Edge Detection. A handful of design parameters from a comparatively large design space are modified to simulate and obtain the results. These parameters are chosen based on their significant effect on the performance and energy consumption of the network and direct impact on network traffic while saving additional simulation costs.

1.6 Thesis Organization

The thesis is organized as follows: In Chapter 2, we discuss the theory behind network on chips and optical network on chips. Additionally, we explore the design of the NoC architectures used for comparison. In Chapter 3, we introduce in detail, the steps involved in the edge detection application used for performing the comparison of the architectures. We also elaborate on the parallelization schemes considered for implementation and the structure

and organization of the bitmap images which are used as workloads for edge detection in this thesis. In Chapter 4, we describe the software implementation of the edge detection application along with the Graphite simulator framework. The experiments for different configurations performed with the multicore architectures and simulation results are presented in Chapter 5. The conclusions are provided in Chapter 6 and the future work is described in Chapter 7.

Chapter 2

Networks on Chip – Components and Architectures

This chapter provides an insight into the components and working of different network on chip (NoC) architectures for multicore systems. The general NoC components are introduced in the context of an electrical and two hybrid opto-electrical NoCs. The hybrid networks discussed in this thesis are clustered architectures: the intra-cluster communication uses electrical network, whereas the inter-cluster network uses an optical ring network. The first section introduces the concept of network on chip and the corresponding components that constitute an on-chip electrical network used for intra cluster communication in the photonic architectures under consideration. It also describes the electrical network on chip architecture considered for the experiments, which uses the electrical mesh topology, which is also the stand-alone, electrical network for our experiments. The knowledge of the operation of intra-chip networks provide an estimate of the various challenges that are present and how the proper choice of network parameters helps in effectively analyzing them. The second section goes on to explain optical network on chip and the additional photonic components applicable for the network communication in this case. The third section introduces the photonic network on chip architectures considered for evaluation.

The necessity of processing data quickly and efficiently has been and is still, a major concern in the field of embedded technology. The processing power offered by a single processor has been exhausted for some time and the attention gradually shifted to the need for multiple processing units on a single chip to magnify the processing power that was provided by the single processing unit[30]. The number of transistors on a chip, which in turn escalates the

processing power, doubles every eighteen months, according to Moore's law[31]**.** This enables addition of more and more processing elements on a smaller die.

Nowadays, with the market supporting many computing systems with multiple processing units has paved way for a new challenge which is having efficient communication between the multiple processing units to solve the computational tasks quickly and efficiently. The coding techniques also require to be revised for parallel execution rather than the conventional serial programming methodology for efficient distribution of tasks among the chip multi processors[32]. Thus, the overall efficiency of an application became dependent on a lot more parameters which are related to the communication network between the processors as well. A consideration of whether the communication between the processing units is going to slow down or speed up the execution of the program compared to its execution with a single or minimal number of processors proves to be vital in choosing the number of processors and the corresponding communication architectures for a particular application**.**

The traditional single core processors and systems with very limited number of cores managed power efficient and optimum bandwidth communication using shared buses for data transfer. This can be accounted to the limited communication requirements in such systems. Ref. [33] identifies scalability as one of the prominent issues with bus based networks, due to the increased bus parasitic capacitance and arbitration complexities in case of larger networks. With the gradual increase in the number of processing cores, bus based communication schemes became tedious and less effective in power efficient, high bandwidth communication [1] and hence on-chip electrical interconnect architectures were introduced [34]. Usage of electrical links as a medium of communication in the network on

chip proved to be sufficient at first, as additional data lines and control lines could easily be implemented by means of single wire connections. Although this would suffice in the case of a smaller system with few processors, it can prove to be highly disadvantageous in the case of a network with a large number of processing units. The scenario of global electrical interconnects in larger networks introduce a case where the propagation delay and interconnect noise increases which in turn affects the bandwidth and ultimately the system performance[12]. The idea of pipelined interconnects for global communication increases the delays and also, the power requirements for the network increases due to higher clock frequency and the requirement of additional registers[12]. The performance-per watt scaling for increasing physical cores in chip multiprocessors is also not addressed efficiently by pure electrical interconnects[3].

The advancements in the field of silicon photonics have provided feasible methods for combining the traditional electrical interconnects with optical interconnects that have high bandwidth and low latency[12]. They also provide bit rate transparency, where the power consumption is independent of the number of bits transmitted. Optical fibers are used as the communication channel in the network and the laser sources are used to produce lights of different wavelengths for communication between different processing units. Micro ring resonators which respond differently to different wavelengths of light act as switches that filter the resonant wavelengths to enable transport of data between the different processing units[35]. Wavelength Division Multiplexing (WDM) technique enables the usage of multiple wavelengths for simultaneous data transmission with minimal losses[36]. This improves the throughput as optical networks use light as the medium of transmission which is much faster than electrical signals and has better immunity to electromagnetic noise[12].

There are different types of architectures proposed for implementation of optical network on chip and with different properties like the physical layout, the number of wavelengths used and the topology used for communication. This thesis is aimed at providing a comparison between the different features of the optical and electrical networks on chip and how it affects the efficiency of the specific application in consideration, with different combinations of network features like the network type, number of processor cores used, the number of clusters, the routing strategy, the number of optical access points, the flit width and the cache properties, used and the workload of the application.

The network on chip architectures considered for the experiments are: one electrical NoC using the mesh topology named Emesh, and two different optical NoCs namely ATAC and ORNoC. They have been modeled on the Graphite [9] multicore simulator which provides a set of configuration parameters that can be used to create different designs.

2.1 Networks on Chip (NoC)

Processing units that are interconnected by means of wires can be categorized based on their communication pattern or topology. The important ones include shared network bus interconnection, point-to-point (crossbars) networks, indirect networks which use NoCs[37].

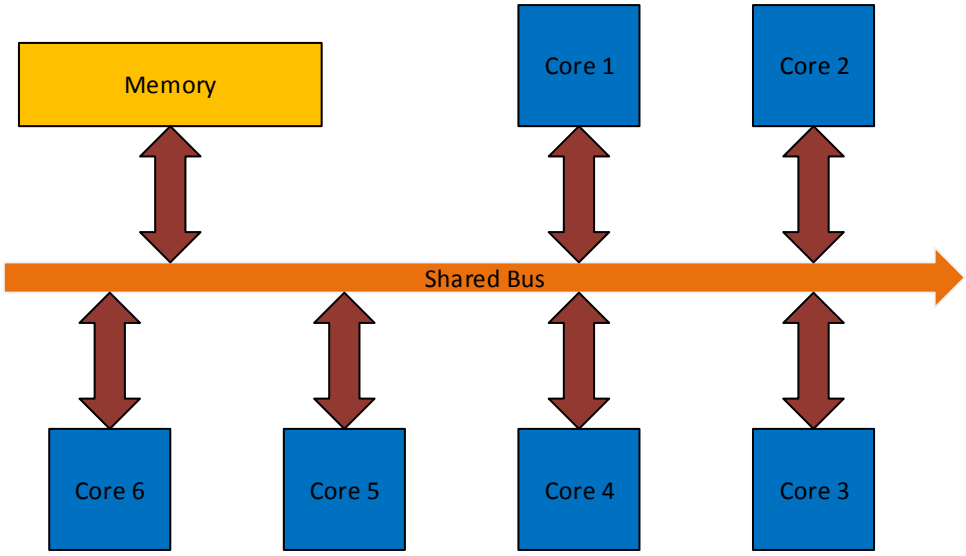


Figure 1: The Shared Bus Architecture

The shared bus architecture uses a shared bus as the name suggests, for communicating between the components in the system. The components follow an arbitration scheme to access control of the bus and transfer data. The main shortcoming of the bus based scheme is the competition for the control of the bus which increases with increase in the number of components connected to it. According to [37], it is also incompetent in addressing the increase in power consumption when more units are attached to the system. Furthermore, it complicates the arbitration process in the cases of buses with multiple masters. The bus based communication structure is depicted in Figure 1.

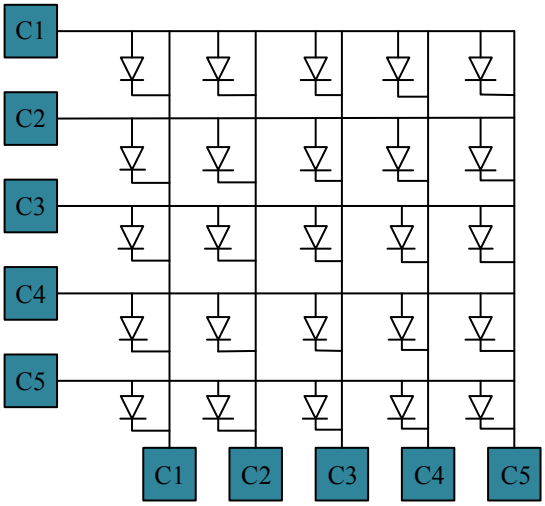


Figure 2: Buffered Crossbar Architecture

The crossbar communication scheme overcomes the contention problem in shared buses and provides good results for smaller networks. The crossbar network involves direct point-to-point connections between all the components in the network. For a smaller network, this provides better connectivity with lesser contention as compared to buses at the expense of more power consumption due to the increased number of connections. The dedicated point-to-point communication topology also introduces improvement in the simplicity of the design and better bandwidth and latency. But, the number of connections increases exponentially with the increasing number of cores. With increasing network size, this leads to increase in the on-chip area consumption and also complicates the wiring layout and routing[37]. The buffered cross bar structure is shown in Figure 2 where C1, C2 etc are the cores and connected by means of tri state buffers.

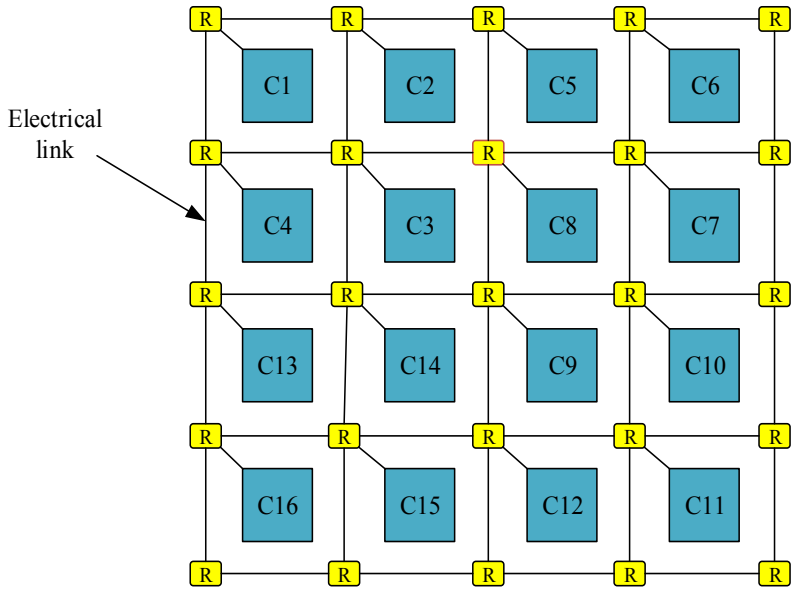


Figure 3: Electrical Mesh (Emesh) architecture

The concept of network on chip provides an organized structure for efficient communication. This implements a network along with proper routing strategies for the specific hardware to reduce the complexity and global foot print. It enables reuse of the components without incurring high scalability costs. It also has lower complexity of design and implementation expense. The electrical network architecture considered in our application is the electrical mesh network which is among the simplest of networks and supported by the simulator. Emesh or electrical networks form a square mesh connecting the cores in the network and uses the XY routing or transfer of data between the cores. Section 2.4.1 explains the XY routing in detail. The electrical mesh network is portrayed in Figure 3, where C1, C2 etc. represent the core numbers and R represents the routers.

According to [38], the major components of any network are links, routers and network interfaces and are described below. The router implements routing strategies based on protocols which are used for transmission of data packets to the next link or the destination core to which they are addressed. The routing protocol has methods laid out mainly to deal

with network contention, to increase throughput and avoid data race situations in the network. The network interfaces establish a connection between the IP cores and the network, which helps in incorporating distinct protocols that each IP might have with respect to the whole network.

Links: The links are represented by wires which form the physical connection between routers in the network. Full-duplex communication enabled networks have dual channels for enabling bidirectional communication simultaneously and the channel bandwidth is represented by the number of wires that constitute the channel. Synchronous as well as asynchronous links can be established in a network based on the bandwidth and throughput requirements.

Routers: Each router needs to be connected to the network as a whole for proper guiding of the packets, a switching matrix for identifying the next router or IP core to pass the packets to and ports to access the IP cores which are connected to it. The routing algorithm defined in a router decides which path the data packet should take to reach its destination. The choice of a routing algorithm depends on the trade-off between the throughput and the latency of the network. The switching scheme emphasizes on how the data transmission takes place, whether through a predefined path by reserving it at the start of the transmission or a more dynamic strategy in which the path to be followed is chosen by the amount of network traffic. When there are multiple requests for data transmission to the same output port, the arbitration policy is responsible for selecting one router input port from which the output port receives the data. When there is a delay in the data transfer, in case of dynamic routing there is a need for the storage of packets temporarily before the network can resume sending them, which is taken care of by the buffering policy.

Network Interfaces: The network interfaces handle the unique protocols that each core has with the network and acts as the boundary between the actual processing and communication. The front end of the network interfaces deal with the requests and acknowledgements for the core while the back end supports the network on chip by mainly ensuring that the packets are encoded and decoded correctly, and synchronizing the data received by the buffers.

2.2 Optical Networks on Chip (ONoC)

Electrical NoCs are advantageous in the case of small-scale networks, which require lesser communication between the cores. With the advent of newer technologies that propose to enhance the processing power of devices, using thousands of cores the shortcomings of the electrical networks are highlighted over their advantages.

Optical networks on chip are introduced as an alternative for efficient communication between multiple cores on a single chip. They make use of silicon photonics for better and effective communication in addition to electrical networks for communication. These ONoCs are offer better scalability with reduced latency and increased bandwidth.

The communication uses a light source and optical waveguides for transmission of data over the optical network. There are optoelectronic interfaces at both the transmitting and receiving ends of the network. The electrical signals are converted to optical signals at the transmitting end before passing it to the network as photons through the waveguide. At the receiving end, a light sensing device like a photodiode detects the photons and they are converted back to the electrical signals and distributed to the respective cores.

2.2.1 Wavelength Division Multiplexing

Wavelength division multiplexing is the optical multiplexing scheme proposed for communication in the optical network on chip architectures in consideration. Different wavelengths of light are multiplexed to be propagated through a single fiber and are de-multiplexed at the receiver end wavelengths. The reduced power requirement for this data transmission scheme also makes it a primary choice for on chip networks.

The number of channels used for communication and the spacing between the wavelengths are directly related. Although there is a theoretical advantage in terms of data transfer rates in having a single channel, multiple channels are often used with proper separation between different wavelengths, to reduce the adverse effects of dispersion and crosstalk. The choice of optimum number of channels required for transmission relies on the trade-off between the speed of transmission needed and the integrity of communication. The imperfections or deformities in the optical fiber while manufacturing also add to the list of losses happening in the waveguide. These losses due to improper manufacturing are common to all media used for transmission.

2.3 Optical Components in ONoC

The integration of silicon and photonics portrays a need for additional components for efficient conversion of signals from electrical to optical domain, transmission and reception and vice versa. Minimal losses in the conversion and transmission processes are necessary to justify the usage of optical network on chip communication architectures. The network is composed of both active and passive optical components that contribute to the electro-optical and opto-electric conversions, their routing and proper detection to regenerate the transmitted electrical signals.

Waveguides: Waveguides are the fundamental unit for optical data transmission. The propagation of optical signals uses the principle of total internal reflection. The difference in refractive induces of the core and cladding components of the waveguide helps guide light through it. Optical waveguide losses need to be minimized for efficient optoelectronic communication. The losses in optical waveguides can be predominantly classified into two – scattering losses occurring due to abnormalities in the core and cladding interfaces during fabrication and absorption losses due to the different bonds in the waveguide material[39]. Such interconnects also undergo process variations due to fabrication imperfections which can result in high losses[39].The bending radius of the optical fibers, and in turn, its cross sectional dimensions needs to be sufficiently large enough to have low bending losses[40]. Ref. [40] also discusses etchless waveguides that can reduce bending losses while allowing for better integration as compared to ridge waveguides.

Lasers: The light sources for communication in the optical interconnects are lasers of multiple wavelengths. Both on-chip and off-chip laser sources are available, each having their respective pros and cons. The off-chip lasers have high light emitting efficiency along with good stability with temperature at the expense of relatively higher coupling losses while integration with an silicon chip[41]. On the other hand, on chip laser sources provide better integration capability, energy efficiency and proportionality with a compact size, with the problem of only the low emission rates of Si limiting the integration process[41]. The important geometric parameters to be considered for the laser sources are: the waveguide distance, the length between the source and the waveguide, the waveguide width, which is the relative difference in radii between the core and cladding, the nominal micro disk laser

radius and the minimum source to source spacing in the case of multiple single wavelength sources.

Modulators: Ring modulators along with the modulator drivers are used for the conversion of electrical signals into optical signals at the opto-electric interface. They also couple the different wavelengths of light into the waveguides at the beginning of transmission in the optical network.

Micro-ring Resonators: The optical switching of different wavelengths of light in ONoCs are carried out by filtering the signals based on the resonant wavelengths by passive optical components called micro ring resonators. They select and redirect the light signals based on their wavelength to the respective destinations. Figure 4 depicts the structure of an optical modulator and the filtering process.

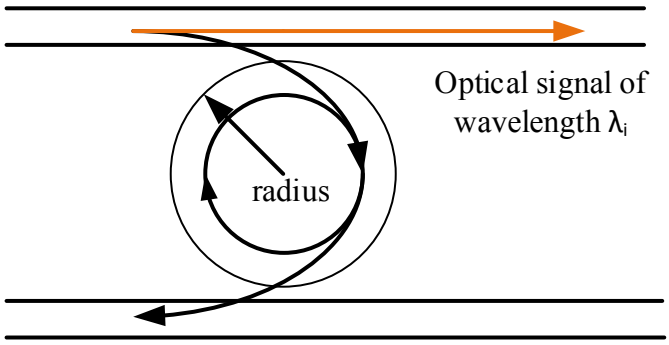


Figure 4: Optical Micro Ring Resonator[42]

The micro ring resonators are sensitive to different wavelengths of light in such a way that they act as filters for coupling a particular wavelength of light into a specific plane of the waveguide relative to the input signal. It works based on the following principle[35]:

- when $\lambda = \lambda_n$, the signal couples into the resonators, and outputs into a waveguide in the same plane as that of the input, where λ_n is the resonant wavelength.
- when $\lambda \neq \lambda_n$, the signal propagates along the same waveguide and the output is obtained on a different plane as that of the input, where λ_n is the resonant wavelength.

Here, signifies the wavelength being transmitted in a given waveguide at a given time, while shows the resonant wavelength for which the transmitted light is passed to the receiver network and finally the destination core.

Photodetectors: The photodetectors form a vital component of the reception part of the ONoC receiver side. The light signals are detected by the photodetectors, which are converted initially to photocurrent based on the wavelength of the light signals used, and transferred to a driver circuit which converts the analog current signals to digital signals. The digital signal is de-serialized and then transferred to the electrical network[43].

2.3.1 Communication Process

The communication process in an ONoC can be broadly classified into intra-layer and inter-layer communications. While the intra-layer communication defines the communication between cores in the *same* electrical layer, the inter-layer communication specifies the communication between *different* electrical layers[43]. The intra-layer communication as per [43] uses purely electrical signals in a 2D mesh architecture with XY routing for communicating between the nodes, where node includes a processor and its local memory.

The inter-layer communication process is slightly more complicated, which uses electrical signals for communicating to the optical network interface (ONI), where the electro-optical conversion of the signals occur followed by the optical routing and the optical signals are received and converted back at the optical network interface and transmitted to the destination node via electrical signals. The optical routing of signals uses the optical

switches mentioned in the previous section to direct the optical signals to the respective ONI. The steps for inter-layer communication[43] is mentioned below:

- **Electro-optical conversion**: this step achieves the conversion by initially serializing the input data and the data is segregated using de-multiplexers. This data is further converted to photonic current by the drivers and used to generate light from the laser. This in turn controls the light intensity of the laser source and assigns specific wavelengths. These wavelengths are then modulated into the waveguides. This procedure is depicted in Figure 5.

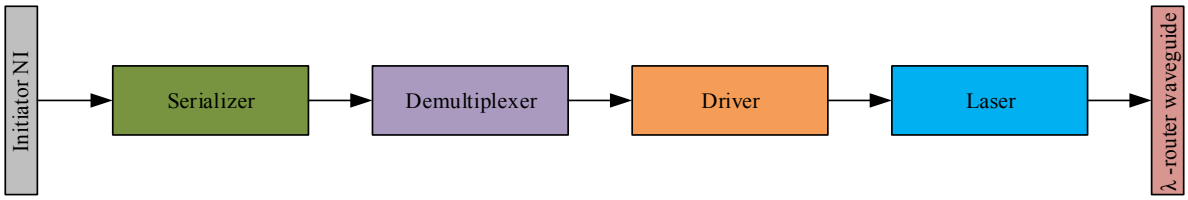


Figure 5: Data transmission in the Electro-Optical Network[35]

- **Optical Routing**: the different wavelengths of light are routed with the help of micro ring resonators, which identifies the path based on the wavelength used, and this process of routing continues till it reaches the destination photodetector.
- **Opto-electrical conversion**: The photodetectors detect the wavelength of light transmitted from the network and converts it into photocurrent and feeds it into a receiver circuit that converts the photocurrent into electrical signals. The data is then de-serialized and stored in buffers before transmitting it to the destination node. Figure 6 represents the steps involved in the reception of optical signals in the network.

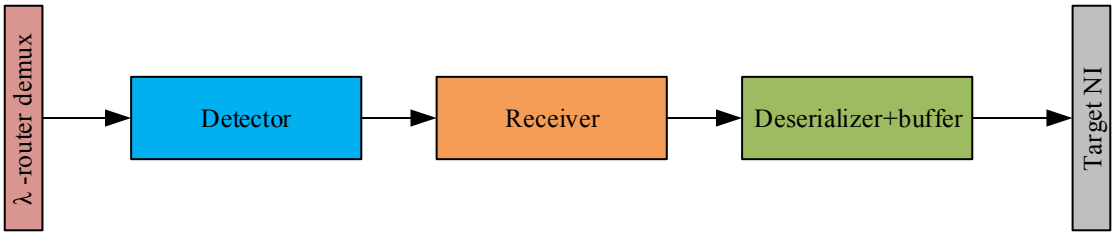


Figure 6: Data reception in the Electro-Optical Network[35]

2.4 Network Architectures Investigated

The architectures used for the experiments are classified into pure electrical and hybrid opto-electrical network on chip architectures. The electrical architecture considered is the 2D mesh based architecture, while the ONoC architectures considered are ATAC and ORNoC. The Emesh architecture considered for simulation is the "Emesh hop-by-hop" scheme in the Graphite simulator, which is the most accurate electrical network modeled in the simulator[44]. Both the ONoC architectures classify the target cores into groups called clusters and they have specific optical hubs, which along with passive and active optical elements handle the transmission of optical signals in the network.

2.4.1 Emesh Network on Chip Architecture

The 2D mesh is one of the simplest electrical network architectures that can be implemented that uses proper routing schemes to minimize the contention delay and bandwidth problems among the other electrical network topologies. The schemes used for the experiments is the "Emesh hop-by-hop" scheme which has the highest accuracy and uses the XY routing scheme, where the packet stops at every intermediate router and is routed based on the current traffic through the network. It also models contention per link in the simulator which would give the estimate of contention delay per link while the others model only the global contention.

The communication pattern is fairly easy to understand as the packets are sent between the source and destination cores through the routers at each intersection. The routers redirect the packets based on the traffic through the network and choose the path with lesser traffic to reach the destination node. In the XY routing scheme, the packets travel initially in the X direction, and after reaching the corresponding column, moves down towards the destination node in the respective column. The Emesh architecture is shown in Figure 3, in section 2.1.

2.4.2 ATAC Optical Network on Chip Architecture

ATAC is an optical network on chip architecture for multi core processors[17] which takes advantage of the optoelectronic technologies for creating contention free network for communication between the processor cores. Optoelectronic technology merged with the current CMOS fabrication processes eliminates the complications with the electrical network which increases with the increasing distance between the physical cores. They also consume lesser power as the optical waveguides have lower losses. It supports multiple cores which are divided into groups called clusters. The optical waveguides forming the optical broadcast network passes through all the clusters forming a loop. The ATAC architecture structure has been represented in Figure 7.

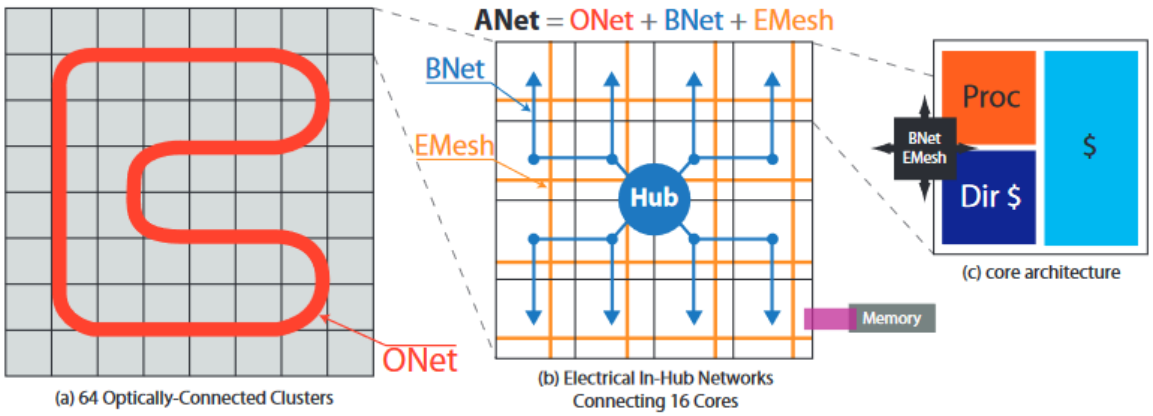


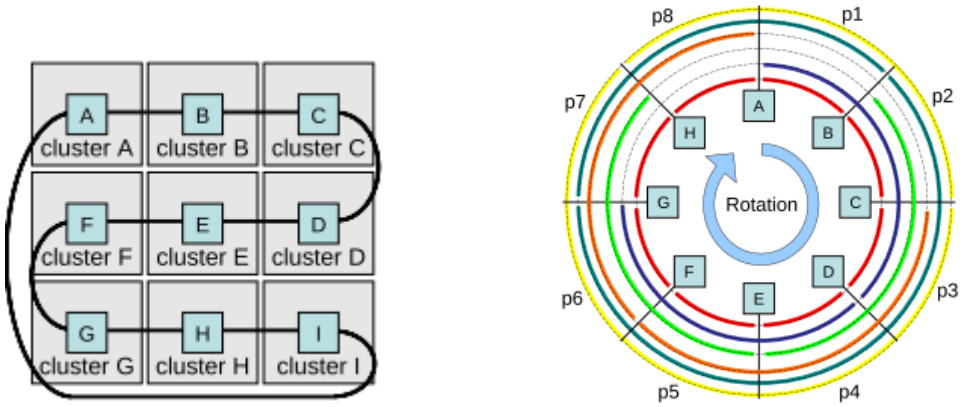
Figure 7: The ATAC Architecture[15]

As clear form the figure, the processor cores in ATAC are connected by two types of networks which are Emesh and ANet. ANet is further divided into multiple types - the optical broadcast network is called ONet, which uses optical hubs and optical waveguides for communication between the clusters. Anet also consists of a couple of electrical networks which are named ENet and Bnet. The Enet transports the data from each of the cores in a cluster to the optical hub and the Bnet transports data from the optical hub back to each of the cores in the cluster.

In ATAC, modulators are used to convert electrical signals into optical signals and off-chip lasers are used as the light source. Ring resonators act as the switches that control the wavelength of light for each cluster as they couple only a specific wavelength of light into the optical waveguide at the transmitting end and selectively couples a particular wavelength form the optical waveguide at the receiving end. Wavelength Division Multiplexing scheme is used to increase the number of wavelengths that can be used for the transmission of bits between the processor cores.

2.4.3 ORNoC Optical Network on Chip Architecture

Optical Ring Network on Chip (ORNoC) [12] is another optical network on chip architecture that also proposes contention free architecture that makes use of the wavelength sharing technique and WDM for highly efficient design. As shown in Figure 8 a), it groups the nodes into clusters which uses electrical network for intra cluster communication and the optical network is used for communication when there is data transfer between two or more clusters. This architecture has an Optical Network Interface (ONI), which provides an interface between the transmitting electrical layer and the optical layer. The receiving end also has an ONI which connects the transmitted data to the receiving electrical interface.



a) b)

Figure 8: a) The ORNoC 2D Architecture [10] b) Wavelength Reuse Implementation [10]

Even though ATAC and ORNoC have the same topology, the fundamental difference between these two architectures is as follows. ATAC has fixed communications, where all optical hubs are connected to all other optical hubs, therefore all optical hubs have the same, large number of modulators for transmission and filters for reception. On the other hand, in ORNoC, the communications are decided during the design time and even if "all-to-all" communication is required, for each configuration (in terms of number of clusters and optical hubs), the communications are implemented using wavelength reuse. Aside from implementing WDM, wavelength reuse in the ORNoC guarantees contention-free communication. The wavelength reuse means that the same wavelength is used for communication between two cores after making sure that the paths for both the transfers do not overlap with each other, as shown in Figure 8 b). ORNoC also uses multiple rings for transmission which makes it scalable and improves power efficiency of the whole system. The optical losses for bidirectional communication in the same waveguide are large, therefore if multiple rings are needed, the ORNoC designer is able to dedicate each ring (i.e. waveguide) to one direction: clock-wise and counter-clock-wise. By using both

communication directions ORNoC can reduce the path that optical signal travels, and therefore the required laser output power that may result in the lower overall power consumption, compared to ATAC. The ORNoC algorithm has 5 fields namely connectivity, ring, portion of ring, wavelength and processed [12]. It assumes that in 2D, there is full connectivity and the value of connectivity field is set to 1. Then the algorithm starts off by assigning a single ring and moves onto more number of rings if required. This is because each ring requires additional power and hence, it is desirable to minimize the number of rings. On identifying the whole path, which is described by sets of ring portions, the unused or reusable wavelengths are selected. The processed field is marked as 1 after finishing the assignment of values to these fields.

Chapter 3

Canny Edge Detection - Application Theory

This chapter describes the Canny edge detection algorithm along with the software parallelization approaches and bitmap images, that are used as workload. The first section describes in detail the steps involved in the Canny edge detection process. The second section portrays the software parallelization schemes considered and the third section justifies the choice of bitmap images as the workload for our application and explains the bitmap file structure.

3.1 Application Used

The Canny edge detection application is chosen based on two important criteria – has widespread usage and also offers enough space for meaningful parallelization. Edge detection is a very significant class of image processing applications, which are vital for the field of computer vision and requires fairly high amount of computation. The quality of edge detection algorithms are judged based on low error rate, the edge point localization on the center of the edge and not detecting fake edges which can be present due to image noise according to John Canny in [8]. The different edge detection schemes have been compared for their efficiency and accuracy in detection in [45] which identifies Canny edge detection scheme as the best amongst them for the aforementioned properties and hence, is chosen to be investigated in this thesis. Canny edge detection performs edge detection through a series of steps which strips down the image into edges.

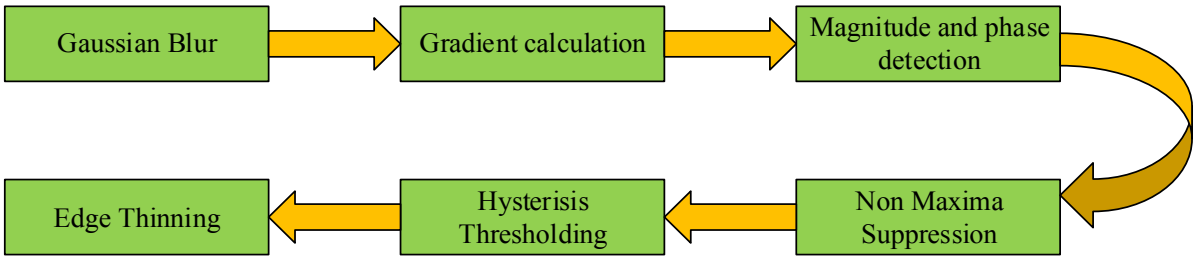


Figure 9: Process flow in the serial Canny edge detection algorithm

The algorithm achieves edge detection through six different stages of processing, which are: (1) Gaussian blur, (2) Gradient calculation, (3) finding the magnitude and phase of the pixels with respect to its neighbours, 4) Non Maxima Suppression (NMS), (5) Hysteresis thresholding and 6) Edge thinning. Figure 9 portrays the steps involved and the flow in the Canny edge detection process. Each pixel has a value for its phase and magnitude from the third stage, which is used for classifying the image pixels into different regions and comparison of magnitude values. Each of the stages in the Canny algorithm[8] are explained below.

3.1.1 Gaussian Blur

Gaussian blur or Gaussian smoothing is employed as the primary stage in edge detection to reduce the image noise and reduce the sharpness and detail. The image is smoothened by averaging out the pixel values with respect to its neighbours. The smoothing process is performed by filtering using a Gaussian kernel, which is a matrix that is used to suppress the high frequency signals and boosts the low frequency signals. After blurring, each pixel value is reassigned a value based on values obtained after convoluting the image pixel with the Gaussian matrix. In Gaussian blur, the Gaussian function is used for establishing the relationship between the pixels. A 3x3 matrix containing Gaussian coefficients is used for blurring the image in this thesis. The extent of blur depends on the standard variation of the Gaussian function. The kernel size is used to set the range of values for the standard

deviation used for Gaussian blur. Detailed information regarding the choice of standard deviation is provided in Section 4.2.

3.1.2 Gradient Calculation

The edge detection occurs after identification of the areas involving sharp variations in the pixels. This necessitates finding out the gradient of the image in the x and y directions. The gradient calculation is performed by convoluting the image pixels with the Sobel kernels[46] in the x and y directions. In [47] the authors compare various image gradient calculation filters and it identifies the Sobel filter as the best one, having minimal mean squared error. This thesis uses 2 3x3 Sobel kernels, which are matrices populated with the discrete differentiation values in the horizontal and vertical directions, for calculating the results.

3.1.3 Magnitude and Phase Calculation

The magnitude and phase for the image is calculated from the image gradients. The magnitude of the pixels is calculated by finding the absolute value of the vector from the horizontal and vertical components, which are the x and y gradients. The alignment of the pixels in the plane is determined by taking the inverse tangent of the image gradient in the y direction with the image gradient in the x direction.

3.1.4 Non-maxima Suppression (NMS)

Based on the range of values to which the calculated phase belongs, each of the pixels are classified into regions and the phase values are resolved into angles along the different directions. The different regions for the phase values are: 1) when the value between 0 and 22.5 degrees or from 157.5 to 180 degrees (region 1), 2) when the value is between 22.5 and 67.5 degrees (region 2), 3) when the value belongs to the range of 67.5 to 112.5 degrees (region 3) and 4) when the value is between 112.5 and 157.5 degrees (region 4). Each of the pixel values that belong to regions 1, 2, 3 and 4 are resolved into 0, 45, 90 or 135 degrees

respectively. The neighbours are identified in Figure 10, where P is the pixel in consideration and the other values represent the regions. This helps in identifying the neighbours along each direction for each of the pixels, which are:

- The ones on the left and right of the pixel for region 1 (R1).
- The ones along the primary diagonal for region 2 (R2).
- The ones to the top and bottom for region 3 (R3).
- The ones along the secondary diagonal for region 4 (R4).

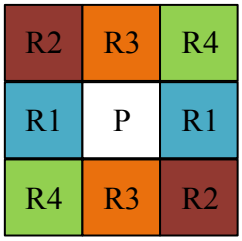


Figure 10: Classification of neighbouring pixels

After this classification of the pixel values into their respective orientation, the magnitudes of the pixel values are compared with the neighbouring pixels along each direction. The pixels with magnitudes lesser than their respective neighbours do not form a local maximum are suppressed, which means they are discarded from further calculation. The pixels with magnitude greater than the neighbours are retained as edge points.

3.1.5 Hysteresis Thresholding

After the non-maxima suppression stage, along with many of the true edge pixels, there can be false pixels due to image noise and variation in colour. The false pixels are removed by checking the magnitude values against a minimum and maximum threshold value to remove the false edges generated mostly due to image noise. The pixel values that are below the minimum thresholds are discarded and the values that are above the maximum threshold are confirmed to be edge points.

3.1.6 Edge thinning

The pixel whose values are in between the minimum and maximum thresholds are judged as edge points based on whether the pixels are connected to solid edge points. The immediate neighbours in all directions are checked for the presence of edge points. The presence of neighbouring edge points indicate that the pixel belongs to an edge and are discarded if they do not belong to the edges.

3.2 Parallelization Schemes

The concept of chip multiprocessors facilitates the use of parallel programming for improving system performance which is most commonly achieved by means of multithreaded programs. Threads are parts of a program that are execute independently from one another which work on same or different data and can execute the same or different instructions concurrently. These methods of instruction level parallelization offered improvements, but rules governing parallelization like the Ahmdahl's law[30] indicate a limitation to achievable speedup. The parallelization schemes utilized in the implementation of the application are pure data parallelization scheme and nested parallelization scheme which uses a combination of task and data parallelization schemes. Traditionally, the types of parallel programming schemes that can be implemented are divided into the following categories as per Flynn's taxonomy: (1) Single Instruction Single Data (SISD), which essentially has no parallelization, (2) Single Instruction Multiple Data which is characterized by the pure data parallelization, (3) Multiple Instruction Single Data, which is non-existent as there cannot be a case where multiple instructions can be executed on a single data concurrently and (4) Multiple Instruction Multiple Data which enables multiple instructions to be executed concurrently on multiple data and has been used along with SIMD and SISD in the mixed implementation.

3.2.1 Pure Data Parallelization Scheme

The pure data parallelization is analogous to the Single Instruction Multiple Data (SIMD) implementation as per Flynn's taxonomy[48]. This type of parallelization strategy executes a single instruction on multiple sets of data. In our application, this involves processor cores executing the same instruction on different blocks of the image and hence provides results much faster compared to the execution of the image by sequential implementation of code. We use user defined block size parameter to divide the entire image width and height into smaller blocks for processing. The quotient of the division with image size as the dividend and block size as the divisor determines the number of blocks in the figure. This block number is used to pass the information of each pixels in a *structure* to the Pthreads. The default block size has been kept as 32 for all the experiments. Figure 11 shows the pure data scheme where PU represents a processing unit, and all the PUs execute the same instructions. We use the term *pure data parallelization scheme* for the implementation that uses only pure data parallelism during all the stages of edge detection.

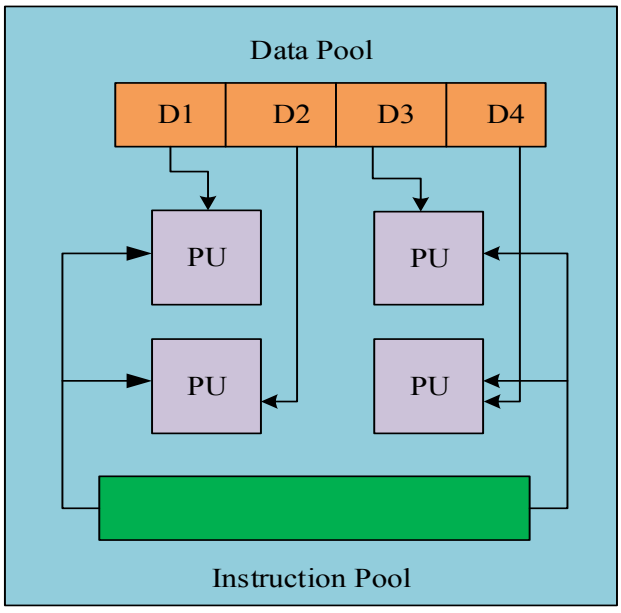


Figure 11: Pure Data Parallelism (a.k.a SIMD)

3.2.2 Mixed Parallelization Scheme

The pure data parallelization scheme uses same parallelization scheme for all the stages as mentioned in the previous section. This section explains mixed parallelization scheme, that uses specific parallelization schemes which are more efficient for each of the stages. Nested instruction and data parallelism, also referred to as MIMD scheme, as per Flynn's taxonomy[48], empowers multiple instructions to be executed on multiple data points concurrently so as to achieve maximum output. It works on multiple blocks of the image and executes multiple instructions on the different blocks of data simultaneously, for our implementation. Figure 12 represents the Nested Instruction and Data Parallelism scheme. We use the term mixed implementation scheme for the implementation that uses pure data parallelism during the Gaussian blur and magnitude and phase calculation stages, nested instruction and data parallelism in the gradient calculation and Non Maxima Suppression stages, sequential implementation for the hysteresis thresholding stage and pure data parallelism for the edge thinning stage.

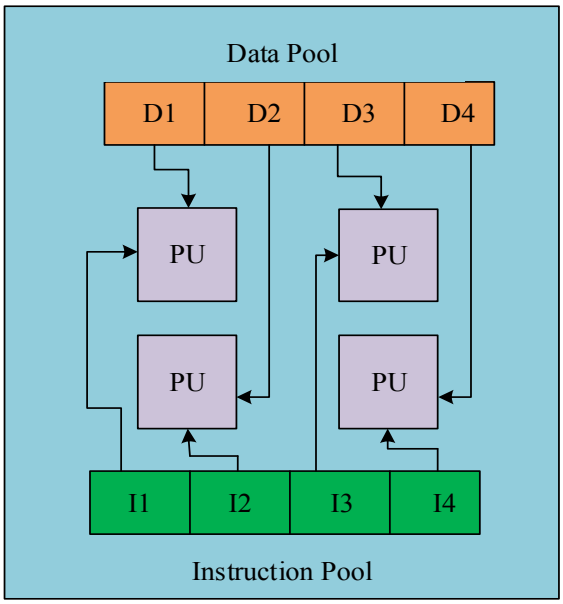


Figure 12: Nested Instruction and Data Parallelism (a.k.a. MIMD)

3.3 Parallelization of the Canny edge detection

The sequential implementation executes all the stages sequentially on the whole image at a time. In this thesis we parallelize Canny Edge Detection application using two schemes. For the pure data parallelism implementation scheme, all the stages use exclusively data parallelism. Similar to the case in [49], the domain decomposition method, which is dividing data into smaller sub sections for processing is adopted for the data parallelism stages. Figure 13Figure 13 shows the process flow in pure data parallelism implementation.

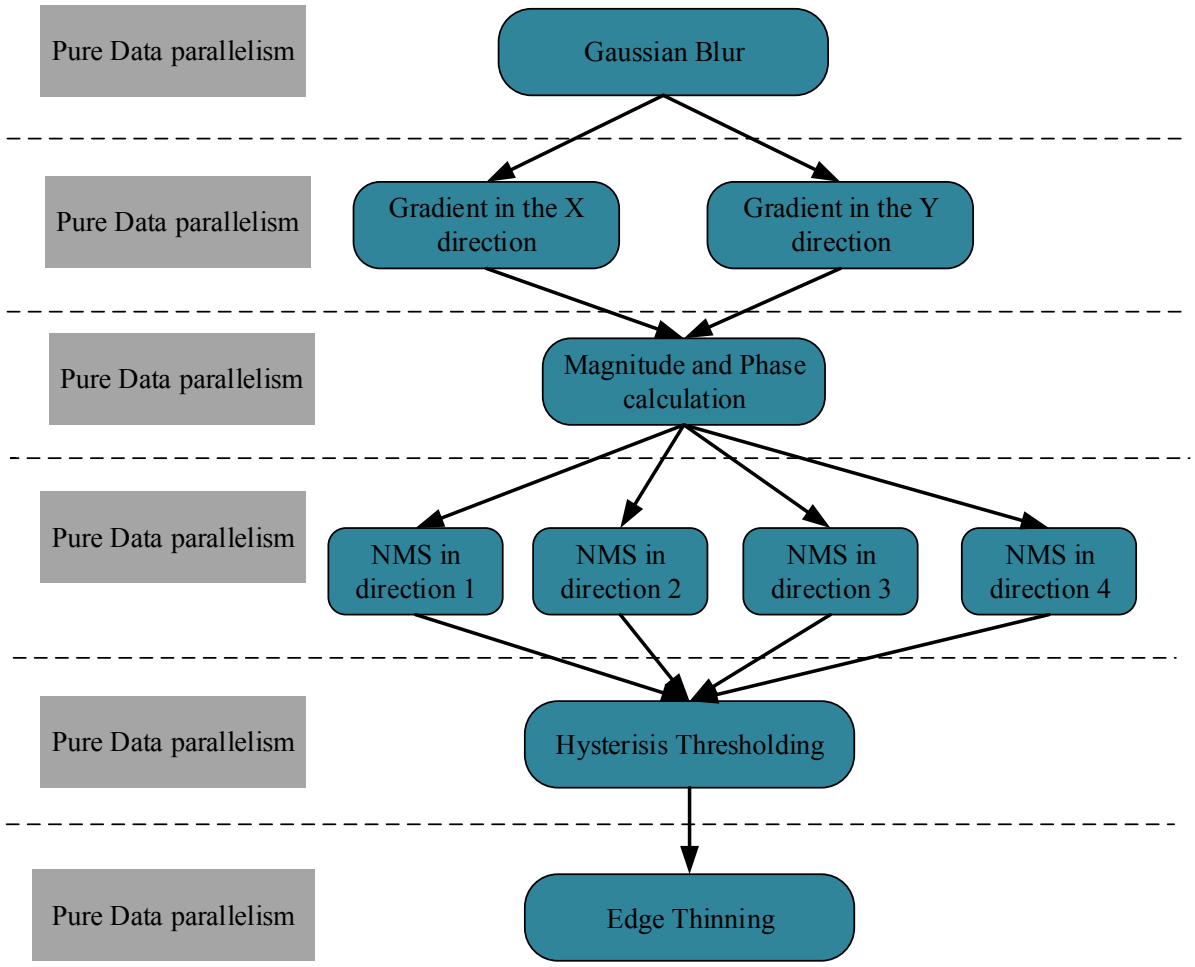


Figure 13: Flow diagram for Pure data Implementation [4]

The details of the mixed implementation are discussed below. The observations from [50] indicate that maximum performance can be achieved for the Canny edge detection application by employing a mix of parallelization schemes for each of the stages. The mixed parallelism implementation scheme combines sequential, pure data and nested instruction and data parallelism.

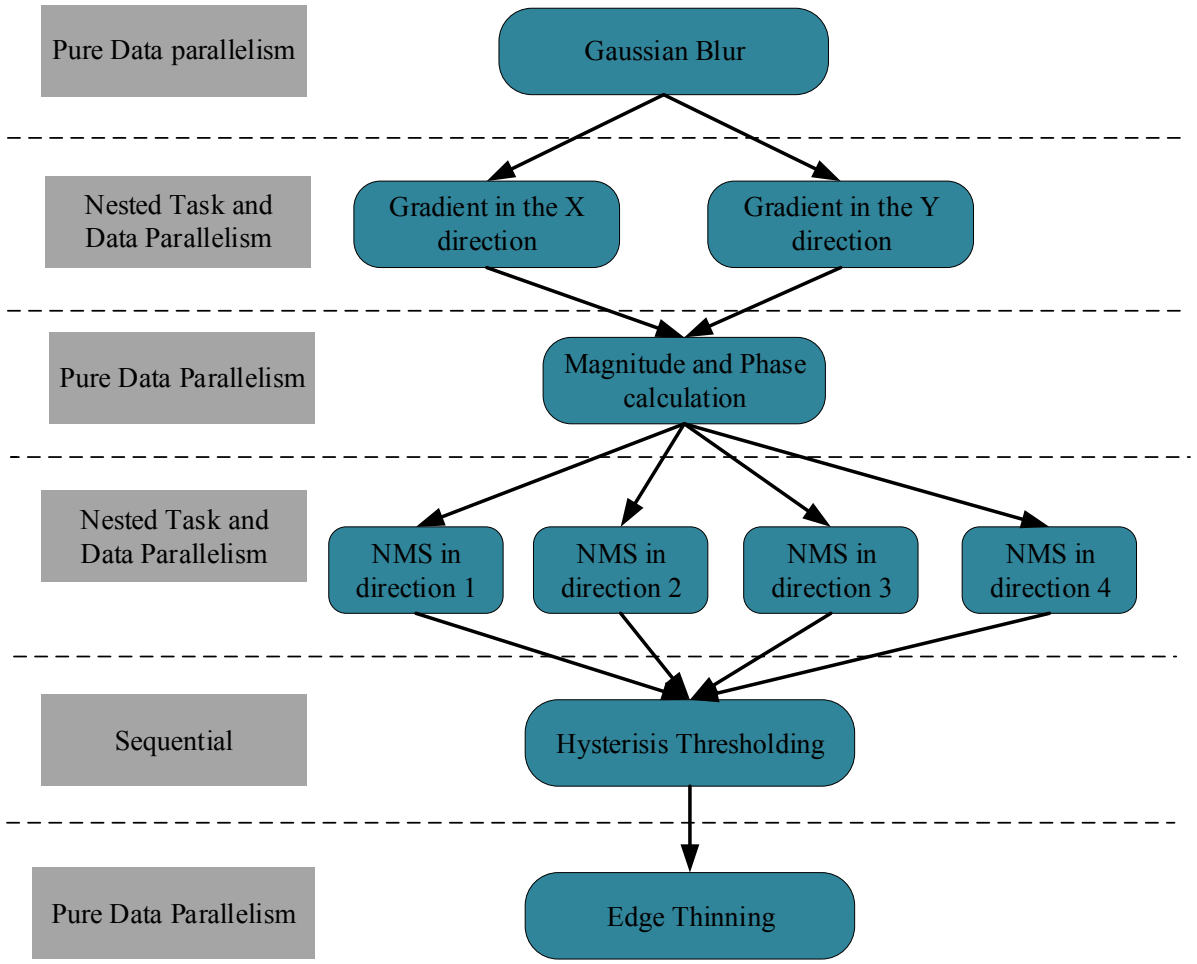


Figure 14: Flow diagram for Mixed Implementation [4]

The mixed parallelism implementation of the Canny edge detection application is depicted in Figure 14, the Gaussian blur stage employs pure data parallelism, in which the image is divided into blocks and the convolution of Gaussian kernel is performed with each of the

image blocks simultaneously. After performing the Gaussian blur, the gradient calculation stage involves nested data and instruction parallelism in which all the different image blocks are convoluted with Sobel matrices in the x and y directions so that the instructions for convolution on the data are processed at the same time. The magnitude and phase calculation stage involves nested instruction and data parallelism as they perform different instructions on the same data blocks simultaneously. As the Non-Maxima Suppression stage involves repeated use of different instructions on different blocks of data, nested instruction and data parallelism is used. The next step involves hysteresis thresholding which is implemented in a sequential manner. The edge thinning uses the pure data parallelism approach.

3.3.1 POSIX Threads

The most common method of implementing parallelization on computing systems is by means of threads. Threads are the smallest sequence of code that executes instructions which are independent from one another, but share a common address space. POSIX threads are defined as an IEEE standard for implementing multithreaded applications. It has a series of predefined functions to perform operations like creation of threads, exiting from a thread function, merging all the created threads together. It also contains functions for avoiding data race conditions, deadlock and livelock situations occurring among threads, using semaphores and mutexes which are harder to implement without the help of pthreads. The deadlock and livelock conditions occur when the threads wait for certain resources. The pthread.h header file that contains all the predefined functions for thread-based operations can be used in many different platforms, but as far the scope of this thesis is concerned, the usage of pthreads only in the C programming language is explored.

3.4 The Bitmap Image

Bitmap image format is one of the easiest image file formats to work with because of its simple encoding scheme. Bitmap images are used in this thesis also because the image load and store operations can easily be performed by the file handling operations in C. In this file format, each pixel can hold a number of bits corresponding to the range of colours that the pixel can represent. The bits per pixel stored in a bitmap image can vary from 1 bit for black and white to 32 bits supporting a variety of colours. The bitmap image has a file structure which consists of four different sections - the bitmap file header, the bitmap information header, the color palette table and the image data in pixel arrays as portrayed in Figure 15.

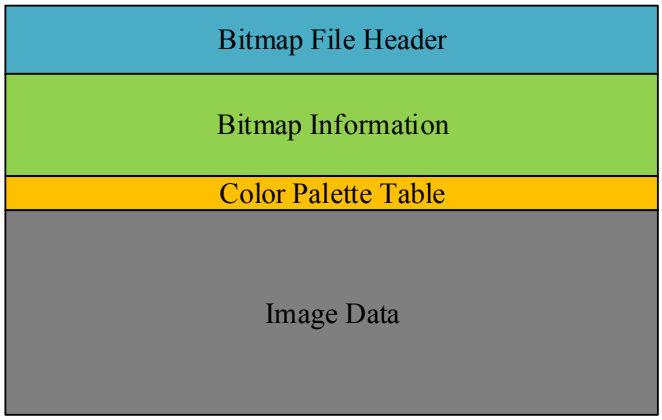


Figure 15: Internal file structure of a bitmap image

3.4.1 Bitmap File Header

The composition of bitmap file header according to[51] is depicted in Table 1. The bitmap file header has a size of 14 bytes. The first 2 bytes are constant values that are the header fields used to identify bitmaps. The pair of ASCII values presented in Table 1 are the values specifying Windows bitmap images which are used in this thesis. The next set of 4 bytes is used to store the size of the bitmap image file. The next 2 locations, each of size 2 bytes are reserved bytes. These reserved values depend on the application used to create the image.

The last set of 4 bytes in the 14 bytes, determines the offset, which is the starting address of the first byte of image data calculated from the bitmap file header.

Table 1: Bitmap File Header[51]

Number of bytes	Information stored
2	0X42 and 0x4D values are stored to identify the header
4	Size of the file in bytes
2	Reserved
2	Reserved
4	Offset bytes

3.4.2 Bitmap Information

The bitmap information header as shown in Table 2, as per [51] contains information regarding image dimensions, colour format and the type of compression used for the bitmap. The initial 4 bytes are used for the storage of the bitmap information header size, which is normally 40 bytes. The next set of 8 bytes, stores the bitmap image width and height in terms of pixels respectively in 4 bytes each. The next 2 bytes store the number of planes, which is 1 in our case. The bits per pixel, which indicates the colour intensities of a pixel is stored in the next 2 bytes. Compression type is indicated in the adjacent 4 bytes and the image size in bytes is stored in the 4 bytes next to it.

Table 2: Bitmap Information[51]

Number of bytes	Information Stored
4	Size of bitmap Info Header
4	Image Width in pixels
4	Image height in pixels
2	Number of planes (always 1)
2	Bits per pixel
4	Compression Scale (0,4,8)
4	Image Size (in bytes)
4	Image Resolution in the X direction (pixel/m)
4	Image Resolution in the Y direction (pixel/m)
4	Number of colour maps used
4	Number of important colours used

3.4.3 Color Palette table

The color palette table section of the bitmap file structure stores information about the blue, green and red colour components for the image. The components depict the intensities of blue, green and red shades in the image considered. The values are retained as zero for the components as this thesis uses grayscale images for the experiments.

3.4.4 Image data

The actual bitmap image data, that is, the pixel values are stored in this section of the bitmap file structure. The values are stored as bytes from left to right in consecutive rows. The pixels are stored form bottom to top in this section of the bitmap file structure. To elaborate, the pixels in the bottom left corner are represented by the first byte and the pixels in the top

right corner is stored in the last byte. In this thesis, the image is used as a two dimensional array as it is the easiest way for division of the image into square blocks for enabling data parallelization.

Chapter 4

Canny Edge Detection Implementation and NoC Simulation

There are 3 different implementations of the Canny Edge Detection (CED) algorithm in the scope of this thesis: the sequential, implementation using pure data parallelism and implementation using the mixed parallelism. The sequential implementation executes the instructions sequentially as the name suggests. While the pure data parallelism uses only data parallelization for all the different stages as the name suggests, the mixed parallelization strategy uses a specific type of parallelization for each stage of the algorithm based on the different operations that need to be performed and the data it works on. The first section explores the various coding platforms which were used for the application development taking into consideration the advantages and disadvantages with respect to our final goal. The second section details the implementation of the bitmap image features and parallelization features. The third section gives a brief idea of the important makefiles used in the simulator. Finally, we describe the NoC simulation framework used in this thesis to run all three implementations of (CED).

4.1 Tools and Coding Platforms

Initially, Matlab was considered for the code implementation, as it is one of the prominent tools for image processing applications and provides plenty of built in functions for processing matrix based applications. Although the implementation is fairly easier in Matlab, the primary concern in this approach is the portability of the application to the simulator. So the approach required translation of the application into C/C++ programming file for

compatibility with the simulator which reduces the code clarity, especially for parallelized portions of the code. The second approach uses OpenCV libraries for implementation of the Canny edge detection application. While it provides built in functions for easily handling image processing operations, the integration of the OpenCV built in libraries with the simulator proved to be difficult which led to the final approach.

The final method implements the code directly in the C programming language with only the basic libraries and built in functions provided by the language. These basic libraries are supported by the simulator unlike those libraries which are user defined in the other two cases.. Even though the initial code development was relatively time consuming, the task of performing experiments and extracting results from the multi core simulator was simplified, as the code execution did not require any specific library integration with the simulator. Bitmap images are used for processing, as mentioned in chapter 3, because of its simple encoding scheme and ease of implementation.

4.2 Implementation

The final approach uses C programming language along with Pthread libraries for implementing the parallelization schemes mentioned in the previous section. The image data is stored and processed in the 2D array format. The Graphite multicore simulator is used to perform the simulation for the specified configurations. The simulator framework distributes the threads for execution on different cores. The information to be processed by the threads and definition of bitmap image features are created using C *structures.* The image data for processing are also organized in *structures* along with other relevant information and are accessed from memory through pointers to the *structure*.

Bitmap features Implementation: The bitmap file structure implementation is necessary for proper loading and saving of the images operations for processing. *Structures* are used to define the bitmap file header, the bitmap information header and the colour palette table mentioned in section 3.5. Pointers are defined for each of the three structures to reference the values during image load and store operations.

The image data is manipulated using file pointers. After loading the image, the pointers to the bitmap features store their respective starting addresses. The file pointer is offset by the value in the bitmap information header so that it points to the address of the image data. In the next step, the pointer that stores the address of the actual image data is invoked, and the image data is read. Vice versa, for storing the image data, the bitmap features mentioned above are filled with appropriate values followed by the corresponding image data by the file pointer.

The images are stored and processed in 2D arrays during all the stages using dynamic memory allocation. 2D arrays make it relatively easier to divide the image into blocks as compared to a 1D array and dynamic memory allocation offers flexible storage of data without memory wastage.

Parallelization Implementation: The application is implemented using sequential, pure data parallelized and mixed parallelized schemes. For all the implementations**,** Graphite distributes the different pieces of code into the specified number of cores mentioned in the configuration file. Parallel programming features are implemented in the application by means of pthreads as mentioned in section 3.3. For dividing the image into smaller blocks, the program uses the block size parameter, which indicates the size of each smaller subsection of the image for applying data parallelism. The number of blocks is obtained by

dividing the image dimensions with the block size (only square images are considered in the experiment to avoid ambiguity). Since multiple image parameters need to be passed to the pthreads for processing, they need to be declared within a *structure* and *structure* objects equal in number to that of the Pthreads are created. To accommodate the rows and columns of each block, the number of threads created is equivalent to the square of the number of blocks.

Since the sequential implementation scheme involves working with the whole image, this does not involve the division of images into smaller blocks. Sequential implementation utilizes the stepwise approach in code implementation. The pure data parallelism implementation utilizes the image division into smaller blocks for processing in each of the stages. The start of the row and column for the image block along with the block size parameter determines the boundaries of the image block. For instance, initially the row and column parameters are set at the beginning of the image and when the block is finished, the parameters are updated with the values that mark the beginning of the next block by adding the block size to the start value. The same procedure continues till the whole image is scaled and has been divided successfully into blocks specified by the user. For the nested instruction and data parallelism, the number of threads created equals the product of number of independent instructions and the number of threads used in pure data parallelism implementation. The number of threads is dependent on the workload, ie, the image size. The number of threads is equal to the square of the quotient of image size in pixels and block size (block size= 32). The implementation of different stages in the Canny edge detection is as shown in [4] mentioned in section 3.1 is explained below.

1) Gaussian Blur: For both the pure data and the mixed parallelization schemes, the Gaussian blur stage is implemented using pure data parallelism while the sequential implementation implements this stage using only stepwise execution. Instruction parallelism is not applicable as there is only one single procedure in this stage. This requires the division of the whole image into smaller blocks. After the image is divided into blocks, each of the blocks needs to be blurred with a Gaussian kernel. The process involves convolution of the image blocks with a matrix formed from Equation 1. A 3x3 matrix is used in the case of our application as it performs greater number of iterations and provides more accurate results compared to matrices of other sizes. The Gaussian kernel is formed by the equation below:

Equation 1: 2D Gaussian kernel equation

$$G(x, y; \sigma) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}$$

where σ is the standard deviation and x and y are the indices of the Gaussian kernel matrix. The standard deviation range is determined by the size of the kernel used. It is determined by the formula in Equation 2.

Equation 2: Kernel Size Calculation

$$n = 2 * \sigma + 1$$

Where n is the kernel size and σ is the standard deviation. The convolution can be approximated to moving the kernel over the image block, multiplying and accumulating the values of the kernel with the image till the end of the block is reached. The sequential implementation performs the convolution on the whole image and does not use separate threads. In both the pure data parallelism and mixed parallelism implementations, Pthreads

equivalent to the square of the number of blocks are used. The blurred image obtained after applying the Gaussian blur operation for a 512x512 image is presented in Figure 16.

Figure 16: Gaussian blurred image with $\sigma = 0.9$

2) Image Gradient Calculation: The second step in the Canny edge detection algorithm is the image gradient calculation. The image gradient calculation determines the differences in the intensity of a pixel along different axes. For a 2D image, the full intensity variation is obtained by calculating the image gradients along the X and Y axes. 3x3 Sobel matrices are used for the image gradient calculation. The sequential implementation performs the convolution on the whole image for both the directions one after the other. This stage is implemented using nested instruction and data parallelism in the case of mixed parallelism implementation, where both gradients are calculated simultaneously, while in pure data implementation the horizontal and vertical gradients are calculated on the blocks of data one after the other and therefore, the former implementation uses more threads. The gradient values of each pixel is calculated by storing and accumulating the values of each iteration into 2 variables, one for each direction and after the iteration completes, they are stored in 2

separate 2 dimensional arrays. The Sobel kernels used for convolution with the image are shown in Equation 3.

Equation 3: Sobel kernels in the X (G1) and Y (G2) directions

$$G1 = \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix} \qquad G2 = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$$

where G1 is the Sobel kernel for calculating gradient along the X-axis and G2 is the Sobel kernel for calculating the gradient along the Y-axis. The resulting images obtained after Gradient calculation along the X and Y directions on the 512x512 pixels standard test image that we use in this work is depicted below, in Figure 17 and Figure 18 respectively. The resultant gradient values form a pair, G_x and G_y, for the gradients along the X and Y axes, respectively.

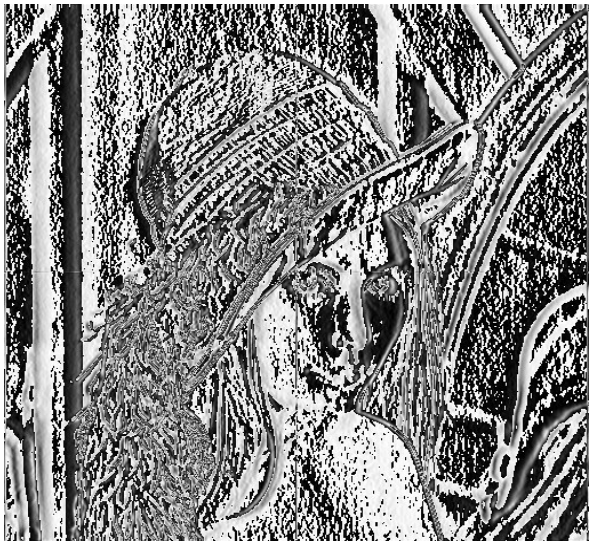


Figure 17: Image gradient along the X direction

Figure 18: Image gradient along the Y direction

3) **Magnitude and Phase calculation:** This stage is executed sequentially in the case of sequential implementation scheme. The magnitude and phase calculation is performed on each block of data simultaneously for mixed parallelism implementation and one after the other for pure data implementation. In pure data parallelism implementation, the number of threads used is the square of the total number of blocks. The number of threads used in the case of mixed parallelism is twice as compared to the number of threads used in data parallelism as there are two separate instructions. The sequential implementation executes the process in a single thread for the whole image. The magnitude calculation involves calculating the magnitude of the image pixels from the x and y gradients. It is calculated as per Equation 4.

Equation 4: Magnitude of the image pixels

$$|G| = \sqrt{{G_x}^2 + {G_y}^2}$$

where G_x is the magnitude the pixel from the 2D array for horizontal gradient values and G_y is the magnitude the pixel from the 2D array for vertical gradient values. This process is

carried out for all the pixels of the image and the magnitude and stored into another 2D array. The image obtained after the magnitude calculation for the gradient pixels is shown in Figure 19.

Figure 19: Magnitude of the image

The phase of the image shows the alignment of pixels and calculated from its x and y gradients. The phase is calculated as the inverse tangent of image gradients in the horizontal and vertical directions. The arctan function is used for image phase calculation and the implementation is based on the following equation.

Equation 5: Phase of the image pixels

$$\emptyset = \arctan\left(G_y / G_x\right)$$

where G_x and G_y are the image gradients in the X and Y directions respectively. The image obtained after calculating the phase is shown in Figure 20.

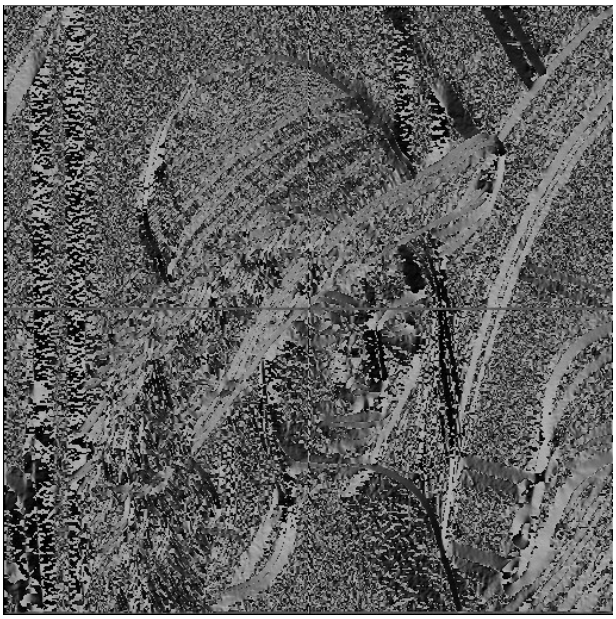


Figure 20: Phase of the image

4) **Non Maxima Suppression:** After resolving the phase of each pixel as mentioned in Section 3.3.4, the magnitude of the pixel retains its value if it is greater than that of the neighbouring pixel in the respective direction and other wise is assigned to 0 in this stage. The sequential implementation performs the conditional checks for the classification into different regions one after the other. The mixed parallelism implementation uses 4 times more number of threads as compared to the pure data parallelism scheme, as the conditional check for each of the regions are performed simultaneously for each block of data while it is performed in a serial manner for pure data parallelism. The sequential implementation performs the conditional verification sequentially for the whole image altogether. The resultant image obtained after performing non maxima suppression on a 512x512 test image is given in Figure 21.

Figure 21: Image after Non Maxima Suppression

5) **Hysteresis Thresholding:** The non-zero magnitude values from the previous stage are compared against a maximum and minimum threshold. The values greater than maximum threshold belongs to an edge, while the ones with values in between are further checked in the next stage for fake edges while the others are discarded. This section of the code is implemented sequentially for both the sequential and mixed data parallelism implementations, while the pure data implementation works on separate data blocks simultaneously. Hence, pure data parallelism implementation uses the same number of threads as in the Gaussian blur stage, while both the other implementations do not use separate threads for this stage. This section also makes use of the conditional statements and logical operators for the evaluation of the expression for all the pixels in the block. Figure 22 shows the image obtained after hysteresis thresholding.

Figure 22: Image after Hysteresis Thresholding

6) **Edge Thinning**: This procedure is aimed at fine graining the edge points by removing the fake edge points that are detected. The pixels with magnitude in between the two threshold values in the previous stage, are compared with neighbouring pixels and based on the results, they are classified to be part of an edge or removed. The mixed parallelism implementation uses a pure data parallelism approach which iterates over different data blocks for the same instruction. The final output image obtained after edge detection is given in Figure 23.

Figure 23: Final image after Canny Edge Detection

The final image obtained after removing fake edges is shown in Figure 23. The image obtained after these six steps contain only the minimal but essential data from the actual image, as edges indicate an area of high intensity variation and hence, are used for processing the next stages of the applications like 3D reconstruction, machine learning and video processing.

4.3 NoC Simulator Framework

Frequently in the design flow, it is desired to develop and optimize software while the hardware is being designed, manufactured and tested. Performing efficient simulations are one of the options for exploring the software techniques for the utilization of various hardware platforms in the development phase. As the technology trends indicate chips with increasing number of cores, the need for performing simulations in order to observe the results and the challenges of the multicore architecture are essential. Even though there are many multicore network on chip simulators like NocSim [23], ORION [24] , PhoenixSim [25], as per the reasons mentioned in section 1.6, the Graphite multicore simulator, proposed in [9], is used for performing the simulations in this thesis. Unlike NocSim, Graphite simulator can model both electrical and photonic NoC architectures. The ineffective router modelling and power modelling techniques used in Orion that also only models electrical NoCs are not suited for the purpose of this thesis. PhoenixSim implements photonic networks while using the electrical network modelling based on the inefficient Orion simulator without any proper interconnection between the two types of networks. Graphite simulator has more effective implementation of both electrical and photonic architectures along with proper efficient interconnects It provides the output values base on a set of input parameters for each simulation, which can be used to quantify the performance of a system

for that specific configuration. It uses the pin tool from Intel for dynamic binary translation, which translates the instructions into the respective binary format for the simulator.

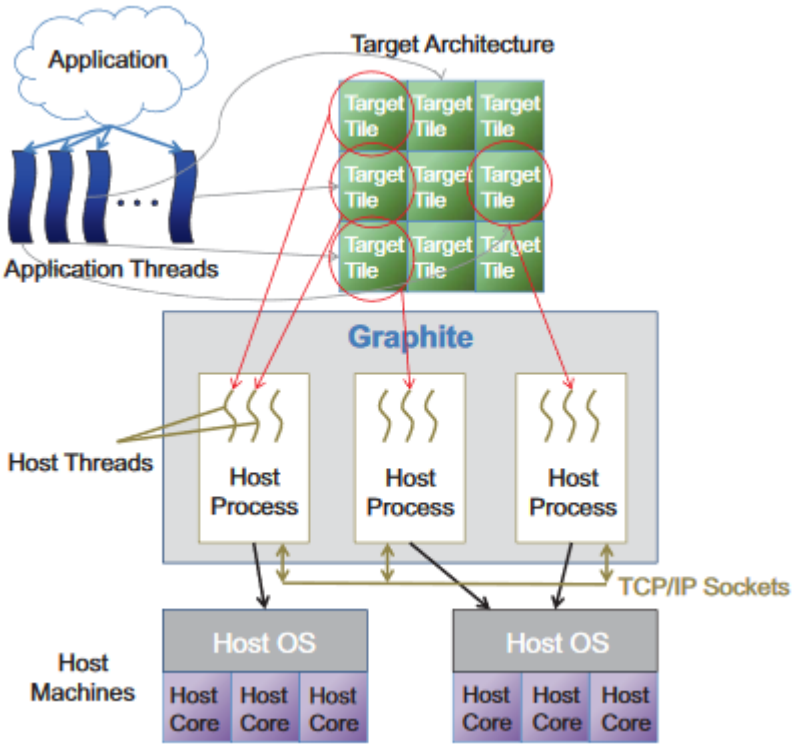


Figure 24: High level achitecture of Graphite simulator [9]

According to Figure 24, the application threads are assigned to a tile of the target architecture and then these threads are distributed across different host processes as host threads which execute on host machine(s). The actual thread scheduling and execution in the host system is then handled by the host operating system[9].

Core modeling: Core modeling handles the instruction fetch and decode units, load and store units as well as execution units in the cores[9] in Graphite. As per [9], the core model is a pure performance model which models the simulated clock exclusive to each tile. It is modelled based on the producer-consumer design, where the different parts of the system

generate information and the model consumes it. It primarily decodes and executes the instructions from the binary translations by the *pin* tool of the instructions from the application threads along with the pseudo instructions for updating the local clock [9]. The main assumption here is a constant cost for instruction execution except for memory and branching operations. Graphite implements an in-order core model, which means that the instructions are fetched, decoded and executed in the order of the request on an out-of-order memory system which stores data in a random order[9].

Network modeling: it describes the modeling of the on chip networks in the simulator. Graphite supports both electrical and optical networks. It supports 5 different types of network models which are categorized as 2 for user-level messages, 2 for shared memory messages and the last one is used for system messages. Our experiments focus only on shared memory networks where the assignment of threads to cores are performed by the simulator and not on message passing where the message transfer between the cores can be directed by the user. Graphite uses synchronization schemes to emulate in order simulation to an extent. Graphite, is not a cycle accurate simulator[52], which means that the simulation is faster but not accurate to the cycles of operation , which means that the simulations can have slight variations in the virtual time of execution. The lax barrier synchronization is used for all the experiments in this thesis, as it also provides the most accurate results compared to the lax and lax p2p synchronization schemes[9]. The lax barrier scheme waits on a fixed number of cycles for all the executing threads to synchronize and this process continues until the task completes execution.

Graphite supports two types of models for electrical mesh networks, which are: Emesh hop counter and Emesh hop by hop. The Emesh hop by hop network is the model considered for

our experiments as it provides the most accurate modelling with a trade-off in performance as compared to the Emesh hop counter model. The Emesh hop by hop model uses XY routing. In this routing scheme, the packet travels along the X direction or horizontally, until it reaches the column containing the destination core and then proceeds with the routing in the Y dimension [52]. Another advantage is that only the hop by hop model offers contention delay modeling for both the user and memory networks, which is one of the output values observed in our experiments.

Power Modeling: it is carried out in Graphite simulator using Design Space Exploration for Network Tools (DSENT) [53]. We trace the static and dynamic energy consumption in the memory network provided by DSENT for our experiments. DSENT can be used in two different instances for power modeling, firstly when an application is being simulated and on the other hand, it is also possible to calculate the standalone power traces for the architectures. We are interested in the former case, as we estimate the total energy consumption in the network while executing the application with a specific set of configuration values. The experiments in this thesis observe the static and dynamic energy consumption in the network. Static power indicates the non-data dependent power which includes the laser power, ring heating and thermal tuning power. The dynamic power includes all the data dependent power like routing data-path, electrical links and receiver networks.

4.4 Makefiles

Graphite predominantly makes use of 3 makefiles for simulating an application:

1) **Configuration makefile**: The Graphite multi core NoC simulator provides an inbuilt configuration makefile with all the different parameters related to network modelling, core

modelling, area modelling and cache modelling. These parameters can be modified by the user to explore the best configuration for each application. These parameters include, but are not limited to the number of cores used and its organization, memory management and distribution schemes, network characteristics, types and properties of the optical components used in the optical networks. Each time a simulation is performed, results are generated based on these configuration parameters and their values. The design space provided by the simulator is enormous and hence, the possible permutations of choices go up to a few thousands of simulation. This is very time consuming and also, all the parameters do not have the same influence on the outputs. Hence, the configuration features are limited to explore only those parameters that are expected to have maximum impact on the outputs observed and those more important to the hardware designers like the latency and power consumption.

2) **Tests Makefile**: it is another general makefile which is provided for executing all the tests. It contains all the actual commands for running the application on Graphite. It also provides the directory paths to invoke the functions which are necessary for performing a successful Graphite simulation.

3) **Application Makefile:** the application makefile is unique for each test application. It provides the application name and target executable name and also permits any specifications in any of the features for this specific test simulation. The application makefile includes the path to the Tests makefile as the Tests makefile performs the actual underlying tasks of compilation, linking and creating the executable for the application.

The simulator stores the result of the simulation in an output file, which stores the output information like energy consumption, latency and contention delay per core, for each simulation for that specific set of configuration parameters.

Chapter 5

Experimental Results

This chapter describes the experimental setup, the parameters modified to extract the results and the different outputs observed as results for the varying network properties in consideration. The first section details the experimental setup and the properties of the modeled network architectures that are varied to observe the results. As far as this thesis is concerned, Graphite simulator is run on a single host machine, with x64 architecture. The experimental results presented in this section are obtained on a quad-core Intel® CPU with 32 GB of DRAM and running at 3.1 GHz. The operating system used is Ubuntu Linux 12.04 and the application is compiled using gcc version 4.6.3. Figure 24 shows the high level architecture of the simulator. The observed results are based on the simulations performed with a specific set of network properties of the different NoC architectures. There are 8 different experiments performed for variation of different network parameters. The parameters that are chosen to be varied for comparison of the network on chip architectures in consideration are presented later in this section for better understanding of the experiments. Each section of experiments is accompanied with the observations and plots from the simulations for the network properties in consideration. The observations and comparisons are of power consumption and delay of the network architectures.

After the implementation of the software parallelized versions of the Canny edge detection application, tests are run to study its performance in a multicore environment with network on chip communication. Graphite[9] multicore simulator executes the application and

collects performance and power statistics. Graphite is not a cycle-accurate simulator, but can perform much faster simulations and also provides synchronization schemes to ensure accuracy of results almost similar to that of cycle accurate simulators[9]. The effects of various properties of the multicore network architectures and their influence on the outputs have been studied to adapt the configuration for optimal performance based on the user constraints. Graphite offers a wide range of configuration capabilities which can be modified to give a lot of experimental cases to observe. The ones discussed in this thesis are those which are deemed very important to analyze the performance of the network. The following sections present our experimental results based on different workloads and NoC design parameters.

Since the simulator offers a large number of parameters that can be changed to observe different results, most of the parameters not in consideration for the experiments are kept as their default values. Even though most of the parameters which are not a part of the actual experiments retain their default settings, some parameters use options different from the default settings like in order to achieve more accurate results. The clock skew management scheme setting has a default option of *lax* scheme which has moderate accuracy and performance, is changed to the *lax barrier* scheme as it provides the most accurate results for the simulations[9]. The lax barrier scheme synchronizes the threads after a pre-defined number of cycles and then proceeds with the execution before the specified cycles are due again and the procedure continues till the end of the application. The application is simulated as a shared memory application where inter thread communication occurs through memory and the shared memory usage is enabled for all the experiments performed.

The experiments featured in this thesis observe the performance of the Emesh hop by hop, ATAC and ORNoC architectures for the Canny edge detection application. The details of these architectures are provided in Section 2.4. The electrical and optical network architectures are compared while varying the image sizes used for edge detection as well as different properties of the network architecture. While the image size variation analyzes the results with the scaling in the size of the image, which is a physical property that does not belong to the network properties, the variation of the number of the cores focuses on finding the influence of the number of cores used for computation when a specific image size is considered. Cluster size defines the number of cores that belong to a cluster and a set of experiments with varying cluster sizes are also performed to observe the trends in performance. The influence of cache properties such as the L1 data cache size and cache line size on the parameters stated above, are also observed. The cache parameters characterize the performance with respect to the data stored in the cache memory as well as the data copied per transfer to the cache from the memory or vice versa. The features which constitute the rest of the experiments are varying flit width, number of optical access points per cluster and the type of routing strategy used. The flit width controls the number of packets in the network while the number of optical communication points in each cluster is determined by the number of optical access points per cluster. The routing strategy primarily indicates the type of routing – the criterion for optical and electrical communication in the network.

The design space for NoCs presents numerous features related to the architectures which can be modified to observe the parameters under observation, but the permutations of the number of possible simulations go up to 50000 or more. Hence, due to the time restriction

for running such a huge number of simulations, the parameters considered for the experiments are the ones which have a direct impact on the network communication for the scope of this thesis.

The results are observed for the average packet latency, average contention delay, total static energy, total dynamic, and total overall energy consumption of the network itself. The parameters that tend to influence energy and delay for a given algorithm are modified to create the experiment set. The experimental setup also involves changing only one of the parameters for each set so that the results obtained demonstrate the influence of only that single parameter. All the experiments are performed for three different software implementations of the Canny edge detection application. The serial implementation involves no parallelization and is the regular sequential implementation of the application. The pure data parallelism approach involves all stages of the edge detection performed on blocks of the image on separate threads. The final implementation approach uses a mixture of different parallelization schemes which has been explained in detail in Section 3.3.

The important configuration parameters used in the experiments and their values used for the experiments are listed in Table 3. The parameters that are varied to observe the results are image size, number of cores in which the algorithm is executed, cluster size, number of optical access points per cluster, flit width, routing strategies, L1 data cache size and the cache line size. All the experiments use the lax barrier synchronization scheme for synchronizing the execution of the threads as mentioned in Table 4.

Table 3: Configuration parameters for the Experiments

Configuration Parameters	Values
Image Size	512/1024/2048 pixels
Number of Cores	16/64/256
Number of Clusters	0/2/4/8
Routing Strategy	Cluster based/ Distance based
Flit Width	16/32/64 bits
Optical Access points per Cluster	0/2/4/8
Cache Line Size	16/32/64/128
L1 Data Cache Size	8/16/32/64/128

Table 4: Default configuration parameters

Configuration Parameters	Values
Synchronization Scheme	Lax Barrier
Image Size	512 pixels – for the cache line sizes in section 5.7 1024 pixels- for all the others
Number of Cores	64
Number of Clusters	4 –for the experiments in sections 5.7 and 5.8 for cache line size and cache size respectively) 8 – for all others
Routing Strategy	Cluster based
Flit Width	64 bits
Optical Access points per Cluster	4
Cache Line Size	32
L1 Data Cache Size	64

The observed results are for average packet latency, network contention delay, total static energy consumption and total dynamic energy consumption of the network. The average packet latency indicates the time taken by a packet to travel from source core to destination core. The average contention delay is a function of the network traffic that includes the

queuing delay for the packet and is updated at each of the routers. It is calculated at the different routers in the target architecture. The ONoC architectures used for the experiments has contention free optical network, so only the electrical network used for intra cluster communication contributes to the contention delay. The static energy consumption indicates the energy consumption due to the data independent components, while the dynamic energy consumption indicates the data dependent energy consumption. The total energy consumption gives the sum of both these energy consumption values. All the relative variations for the performance parameters with respect to Emesh is computed with Equation 6.

Equation 6: Equation for Relative Variation

$$ONoC\ (relative\ value) = \frac{Emesh(value) - ONoC(value)}{Emesh(value)}$$

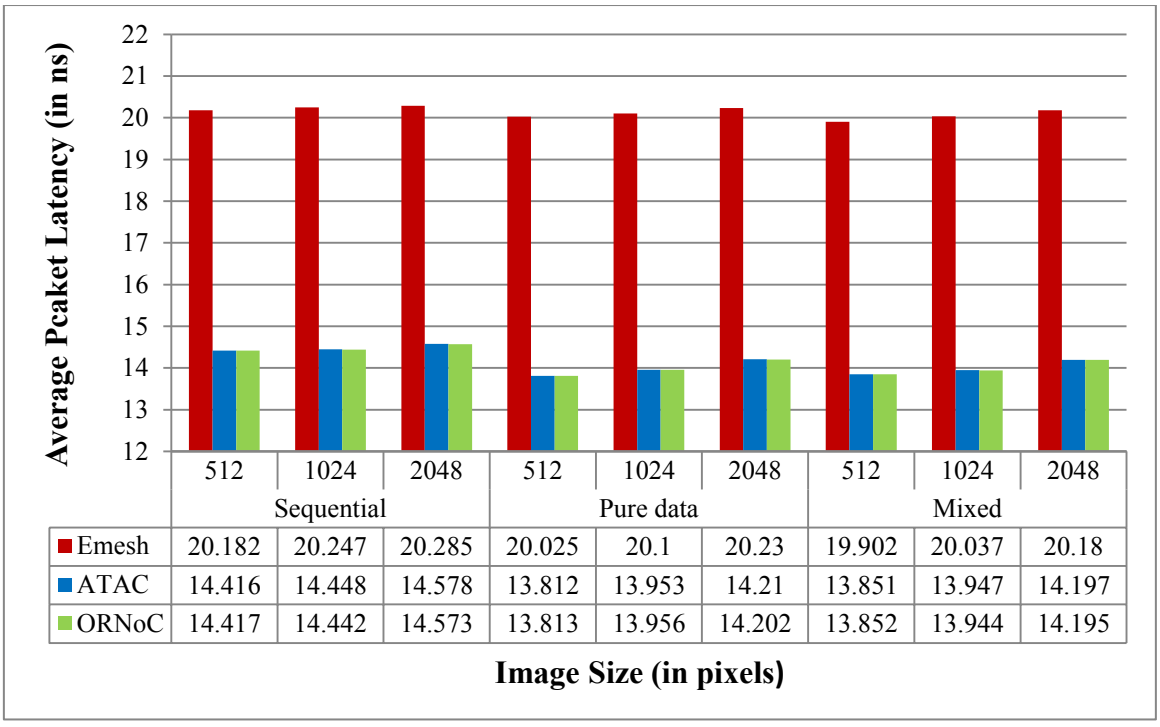
5.1 Analysis with varying Image sizes

The first set of experiments analyzes the network features with different input image sizes. This studies the impact of increased data load on the processing cores and the network. The images considered for this experiment are of 512 x 512, 1024 x 1024 and 2048 x 2048 pixel square images. The corresponding file size is 237 KB, 1 MB and 4 MB, respectively. The images bigger than this take a very large time for simulation and also, the algorithm is not very efficient when applied unless the image is compressed to reduce the size. For example, the size of a 4096 x 4096 square image is 16.8 MB which is too big and needs to be compressed before processing for the efficient use of the algorithm. The implementation and parallelization of Canny Edge Detection algorithm differs significantly if applied to

compressed image, and it is out of the scope of this thesis. The rest of the configuration parameters are kept as default per Table 4.

The results are evaluated for both electrical and optical NoC architectures. The value of the number of cores is kept as a constant, at 64 and the cluster size for the optical networks is kept as 8 for the scope of this experiment. The detailed configuration parameters are provided in **Error! Reference source not found.**. The number of cores is selected as 64 as it is the moderate core size which is not too high or not too low for all the workloads under consideration for this experiment, with a cluster size of 8 as it involves more communication in the optical network.

5.1.1 Results for Packet Latency for varying Image Sizes



	Sequential			Pure data			Mixed		
	512	1024	2048	512	1024	2048	512	1024	2048
Emesh	20.182	20.247	20.285	20.025	20.1	20.23	19.902	20.037	20.18
ATAC	14.416	14.448	14.578	13.812	13.953	14.21	13.851	13.947	14.197
ORNoC	14.417	14.442	14.573	13.813	13.956	14.202	13.852	13.944	14.195

Figure 25: Average Packet Latency for Image Sizes NxN, N = {512, 1024, 2048}

Latency increases with the increase in image sizes, because of the increase in workload which necessitates the processing of more packets which causes more traffic in the network. The results are as expected as both the ONoC architectures use wavelength division multiplexing scheme which ensures faster data transfer, with ORNoC having the additional feature of wavelength reuse. The Emesh architecture requires more time . For the Emesh architecture, the packet latency increases from 20.182ns when the image size is 512 x 512 pixels, to 20.285ns for 2048 x 2048 pixels in the case of sequential implementation, while it is between 20.025ns and 20.23ns for pure data parallelism implementation for 512 pixel square image and 2048 pixel square image respectively and the values are between 19.902ns for 512 pixel square image and 20.18ns for the 2048 pixel square image for the mixed implementation. The ATAC architecture shows packet latency values from 14.416ns when the image size is 512 x 512 pixels to 14.578ns for 2048 x 2048 pixels in the case of sequential implementation, while it is between 13.812ns and 14.21ns for pure data parallelism implementation for 512 pixel square image and 2048 pixel square image respectively and the values are between 13.851ns for 512 pixel square image and 14.197ns for the 2048 pixel square image for the mixed implementation. For the ORNoC architecture, the packet latency increases from 14.417ns when the image size is 512 x 512 pixels, to 14.573ns for 2048 x 2048 pixels in the case of sequential implementation, while it is between 13.813ns and 14.202ns for pure data parallelism implementation for 512 pixel square image and 2048 pixel square image respectively and the values are between 13.852ns for 512 pixel square image and 14.195ns for the 2048 pixel square image for the mixed implementation.

The average comparison includes the results for all the 3 image sizes. The average packet latency values are 20.238ns for Emesh, 14.481ns for ATAC and 14.477ns for ORNoC in the case of sequential implementation. The pure data parallelism implementation has average values of 20.118ns, 13.992ns and 13.99ns for Emesh, ATAC and ORNoC architectures respectively. The mixed implementation shows average values of 20.04 for Emesh, 13.998ns for ATAC and 13.997ns for ORNoC. The average improvement for all the 3 image sizes in the cases of pure data parallelism and the mixed implementation schemes against the sequential implementation for all the image sizes are 0.59% and 0.98% respectively for Emesh, 3.38% and 3.36% respectively for ATAC, 3.36% and 3.32% respectively for ORNoC architectures.

. The efficient parallel processing of data improves the latency values of parallelized schemes as compared to the sequential implementation. The mixed parallelization shows better values than pure data parallelism because of more effectiveness in the parallelization scheme.

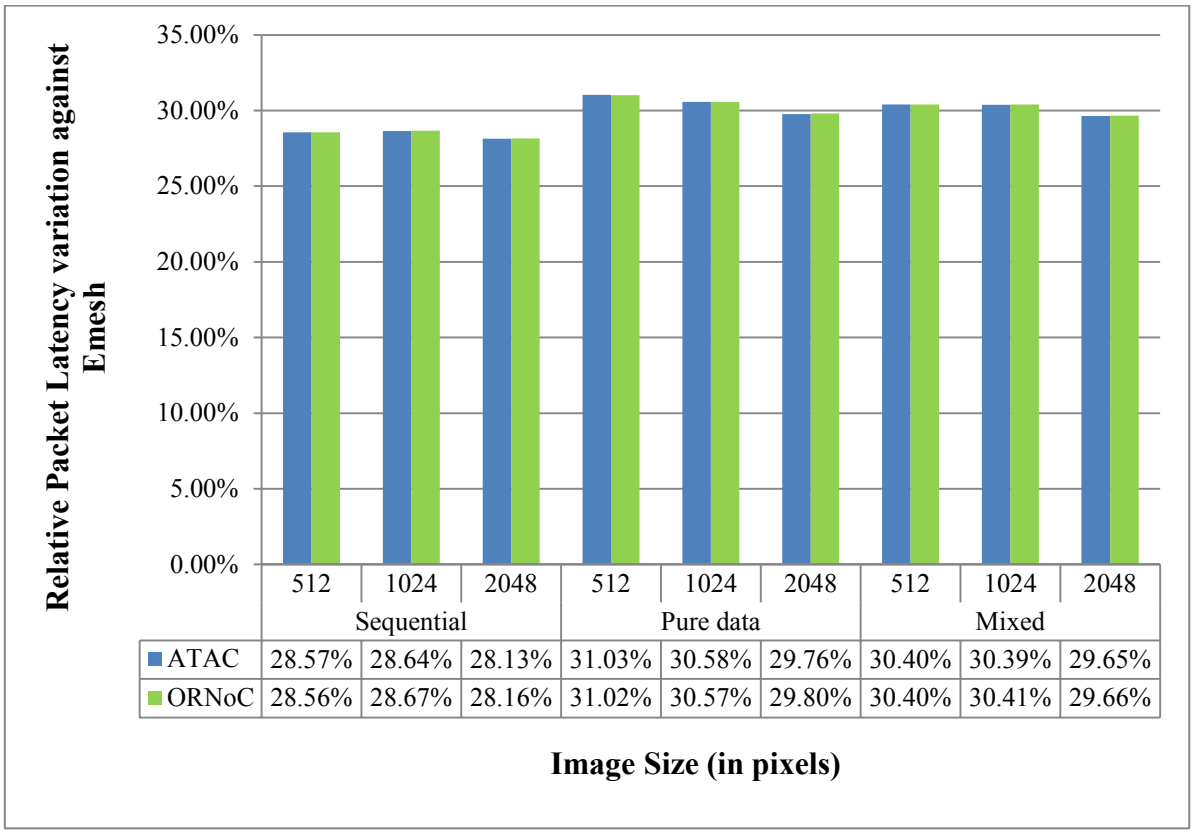
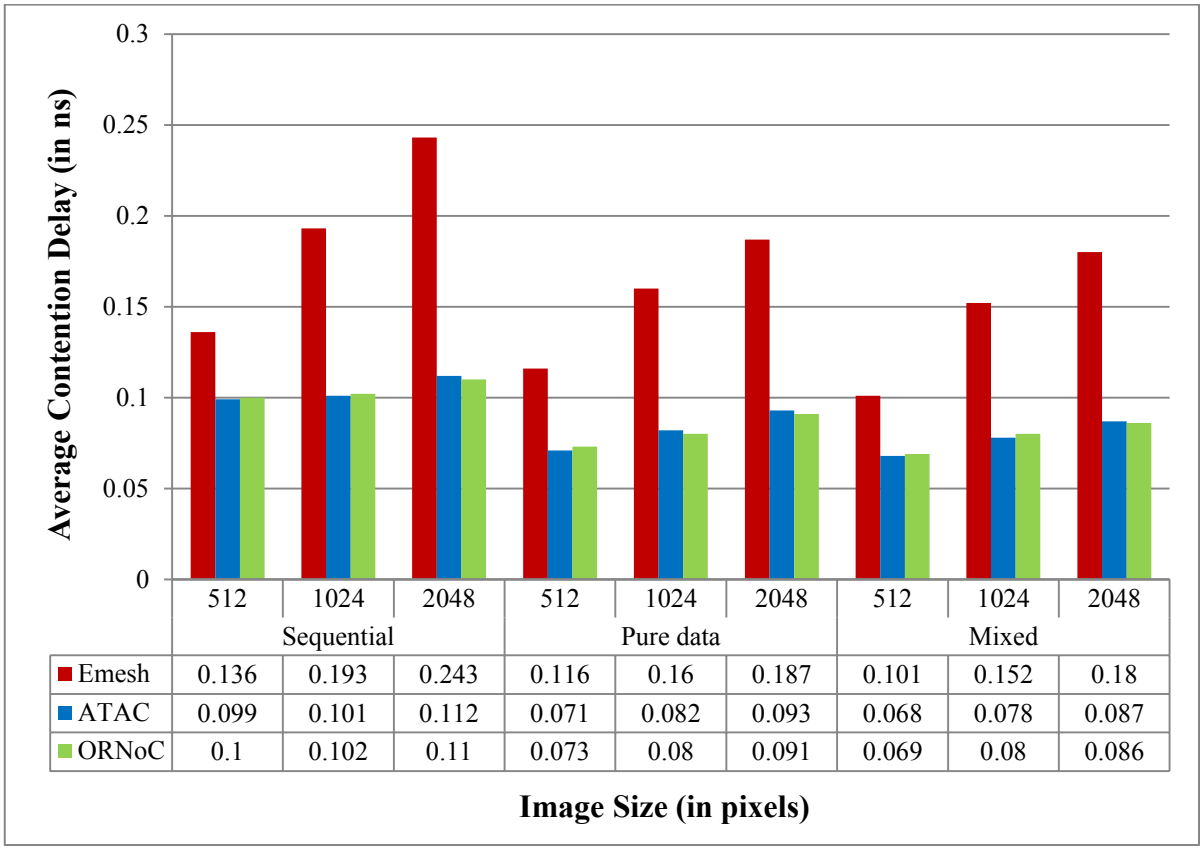


Figure 26: Relative Latency variation for Image Sizes NxN, N = {512, 1024, 2048}

The relative latency variation values against Emesh for ATAC range from 28.57% for 512 pixel square image to 28.13% for 2048 pixel square image in the case of sequential implementation, 31.03% to 29.76% in the case of 512 pixel square image and 2048 pixel square image respectively for pure data parallelism implementation and 30.4% to 29.65% in the case of mixed parallelism implementation for 512 pixel square image and 2048 pixel square image respectively. The relative latency variation values for ORNoC range from 28.56% for 512 pixel square image to 28.16% for 2048 pixel square image in the case of sequential implementation, 31.02% to 29.8% in the case of 512 pixel square image and 2048 pixel square image respectively for pure data parallelism implementation and 30.4% to 29.66% in the case of mixed parallelism implementation for 512 pixel square image and 2048 pixel square image respectively. The shows that the average improvement in relative

latency values for all the three image sizes in the case of sequential implementation is 28.45% for ATAC and 28.46% for ORNoC while the pure data parallelization implementations show a marginal increase of 30.46% and 30.46% for ATAC and ORNoC on the average, respectively. The average values for all the 3image sizes in the case of mixed parallelism implementation are 30.15% and 30.16% for ATAC and ORNoC architectures respectively. Both ONoC architectures perform better compared to Emesh as clear from the graph due to efficient communication enabled by the optical network. The relative improvement in the case of both the ONoC architectures is very close to each other.

5.1.2 Results for Contention Delay for varying Image Sizes



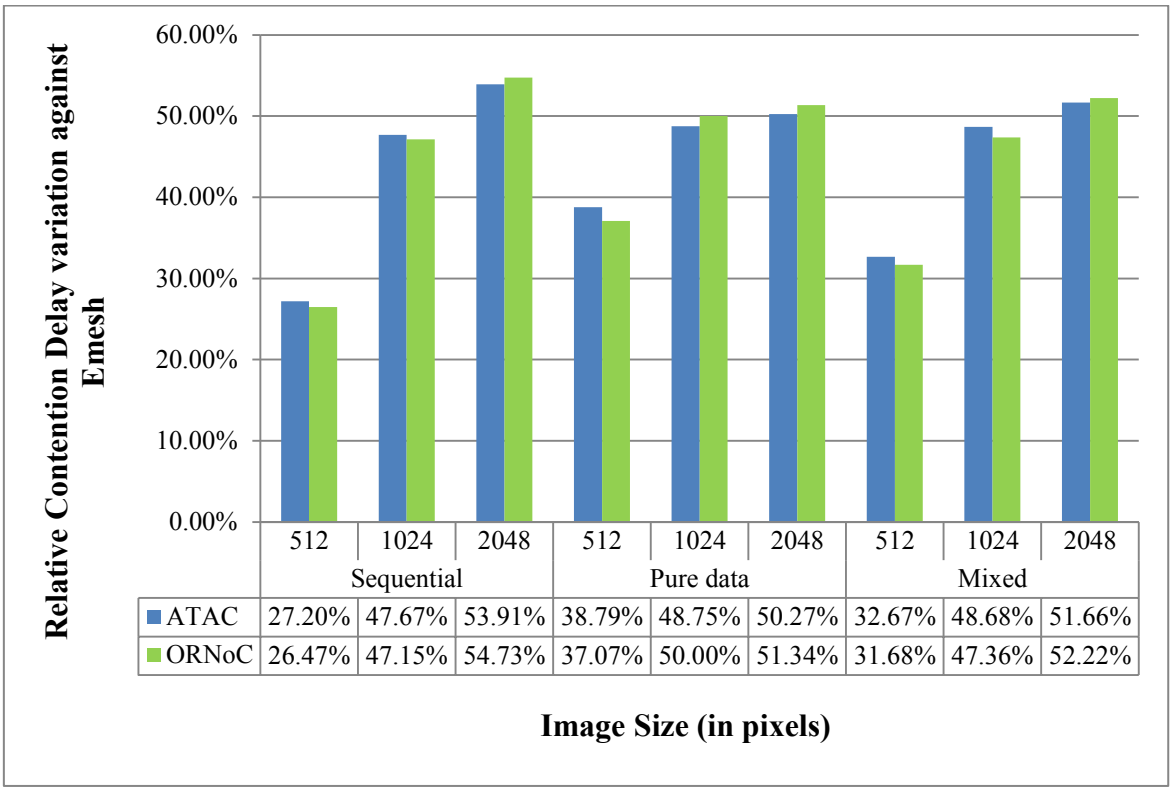
	512	1024	2048	512	1024	2048	512	1024	2048
	Sequential			Pure data			Mixed		
Emesh	0.136	0.193	0.243	0.116	0.16	0.187	0.101	0.152	0.18
ATAC	0.099	0.101	0.112	0.071	0.082	0.093	0.068	0.078	0.087
ORNoC	0.1	0.102	0.11	0.073	0.08	0.091	0.069	0.08	0.086

Figure 27: Average Contention Delay for Image Sizes NxN, N = {512, 1024, 2048}

The contention delay is also observed to increase with image size, similar to the reasoning for latency values, due to the larger workload creating more network traffic in the electrical network which contributes for the increase in the case of all the three architectures. The increase in contention delay values for the Emesh architecture are from 0.136ns when the image size is 512 x 512 pixels to 0.243ns for 2048 x 2048 pixels in the case of sequential implementation, while it is between 0.116ns and 0.187ns for pure data parallelism implementation for 512 pixel square image and 2048 pixel square image respectively and the values are between 0.101ns for 512 pixel square image and 0.18ns for the 2048 pixel square image for the mixed implementation. The ATAC architecture shows the contention delay values from 0.099ns when the image size is 512 x 512 pixels to 0.112ns for 2048 x 2048 pixels in the case of sequential implementation, while it is between 0.071ns and 0.093ns for pure data parallelism implementation for 512 pixel square image and 2048 pixel square image respectively and the values are between 0.068ns for 512 pixel square image and 0.087ns for the 2048 pixel square image for the mixed implementation. For the ORNoC architecture, the contention delay values increase from 0.1ns when the image size is 512 x 512 pixels to 0.11ns for 2048 x 2048 pixels in the case of sequential implementation, while it is between 0.073ns and 0.091ns for pure data parallelism implementation for 512 pixel square image and 2048 pixel square image respectively and the values are between 0.069ns for 512 pixel square image and 0.086ns for the 2048 pixel square image for the mixed implementation.

The average values of contention delay are calculated for all the three image sizes. The average contention delay values are 0.191ns for Emesh and 0.104ns for both ATAC and ORNoC architectures in the case of sequential implementation. The pure data parallelism

implementation shows average values of 0.154ns, 0.082ns and 0.081ns for Emesh, ATAC and ORNoC architectures respectively. The average contention delay values for the mixed implementation scheme are 0.144ns for Emesh and 0.078ns for both ATAC and ORNoC architectures. The average improvement in contention delay for all the 3 image sizes is 19.37% for the pure data parallelism and 24.61% for the mixed parallelism implementation in the case of Emesh architecture. The average improvement rate for all the 3 image sizes against sequential implementation are 21.15% for pure data parallelism implementation and 25% in the case of mixed parallelism implementation for ATAC while the pure data parallelism shows average improvement rate of 22.11% and the mixed parallelism implementation shows 25% average improvement rate for ORNoC. The sequential implementation scheme has higher contention delay, while both the parallelized schemes have values better than the sequential scheme. The parallelized implementations have values very close to each other for the optical architectures, while they show notable difference between the pure data and mixed parallelization schemes for Emesh with the mixed parallelization scheme outperforming the pure data parallelization scheme. The improvement in mixed parallelism implementation over pure data parallelism can be accounted to the efficient implementation of parallelization strategies in the mixed implementation.



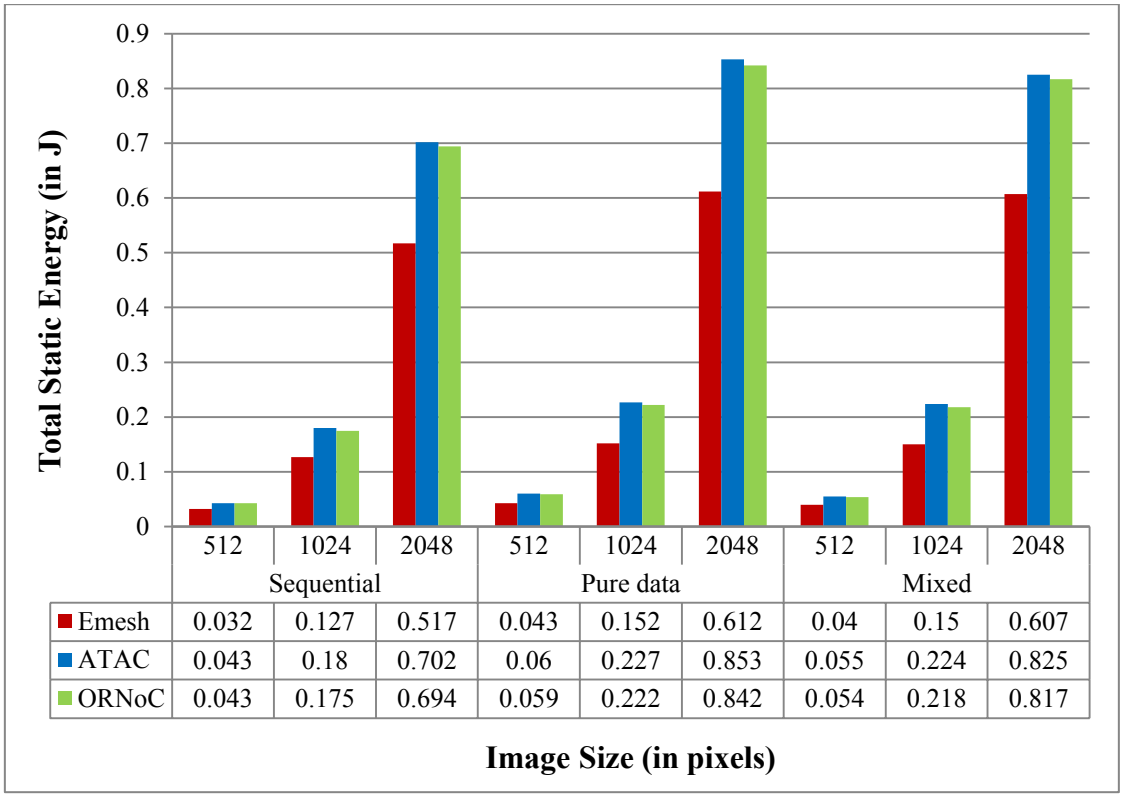
	512	1024	2048	512	1024	2048	512	1024	2048
	Sequential			Pure data			Mixed		
ATAC	27.20%	47.67%	53.91%	38.79%	48.75%	50.27%	32.67%	48.68%	51.66%
ORNoC	26.47%	47.15%	54.73%	37.07%	50.00%	51.34%	31.68%	47.36%	52.22%

Figure 28: Relative Contention Delay variation for Image Sizes NxN, N = {512, 1024, 2048}

Figure 28 shows the improvement in latency for the ONoC architectures against Emesh. The relative contention delay values for ATAC range from 27.2% for 512 pixel square image to 53.91% for 2048 pixel square image in the case of sequential implementation, 38.79% to 50.27% in the case of 512 pixel square image and 2048 pixel square image respectively for pure data parallelism implementation and 32.67% to 51.66% in the case of mixed parallelism implementation for 512 pixel square image and 2048 pixel square image respectively. The relative latency variation values for ORNoC range from 26.47% for 512 pixel square image to 54.73% for 2048 pixel square image in the case of sequential implementation, 37.07% to 51.34% in the case of 512 pixel square image and 2048 pixel square image respectively for pure data parallelism implementation and 31.68% to 52.22% in the case of mixed parallelism

implementation for 512 pixel square image and 2048 pixel square image respectively. The average improvement is calculated for all the 3 image sizes. The average improvement in the case of sequential implementation is 42.92% and 42.78% for ATAC and ORNoC respectively, in the case of pure data parallelism implementation are 45.94% for ATAC and 46.14% for ORNoC and in the case of mixed parallelism implementation are 44.34% and 43.75% for ATAC and ORNoC respectively. As observed from the above figure, the contention delay for ATAC and ORNoC are better compared to Emesh architecture due to the presence of optical network. ORNoC and ATAC architectures have contention delays in close proximity.

5.1.3 Results for Static Energy for varying Image Sizes



	512	1024	2048	512	1024	2048	512	1024	2048
	Sequential			Pure data			Mixed		
Emesh	0.032	0.127	0.517	0.043	0.152	0.612	0.04	0.15	0.607
ATAC	0.043	0.18	0.702	0.06	0.227	0.853	0.055	0.224	0.825
ORNoC	0.043	0.175	0.694	0.059	0.222	0.842	0.054	0.218	0.817

Figure 29: Total Static Energy for Image Sizes NxN, N = {512, 1024, 2048}

The increase in workload would average the usage of all optical components for longer duration of time which is reflected by the increase in static energy consumption. For the Emesh architecture, the static energy increases from 0.032J when the image size is 512 x 512 pixels to 0.517J for 2048 x 2048 pixels in the case of sequential implementation, while it is between 0.043J and 0.612J for pure data parallelism implementation for 512 pixel square image and 2048 pixel square image respectively and the values are between 0.04J for 512 pixel square image and 0.607J for the 2048 pixel square image for the mixed implementation. The ATAC architecture shows the static energy values from 0.043J when the image size is 512 x 512 pixels to 0.702J for 2048 x 2048 pixels in the case of sequential implementation, while it is between 0.06J and 0.853J for pure data parallelism implementation for 512 pixel square image and 2048 pixel square image respectively and the values are between 0.055J for 512 pixel square image and 0.825J for the 2048 pixel square image for the mixed implementation. For the ORNoC architecture, the static energy values increase from 0.043J when the image size is 512 x 512 pixels to 0.694J for 2048 x 2048 pixels in the case of sequential implementation, while it is between 0.059J and 0.842J for pure data parallelism implementation for 512 pixel square image and 2048 pixel square image respectively and the values are between 0.054J for 512 pixel square image and 0.817J for the 2048 pixel square image for the mixed implementation. From the results, the pure data parallelism implementation consumes more static energy as works on subdivided image blocks on all the stages which requires more energy which is data independent while the mixed implementation uses a mixture of parallelization schemes which has a sequential implementation stage that does not require splitting the image and hence, consumes lesser energy as compared to pure data implementation.

The average value static energy consumption of is calculated for all the 3 image sizes. The average values for static energy consumption for Emesh architecture are 0.225J, 0.269J and 0.266J respectively for the sequential, pure data parallelism and mixed parallelism implementations. The results for ATAC show average values of 0.308J, 0.38J and 0.368J for sequential, pure data parallelism and mixed data implementations. The results for ORNoC show results which have average values of 0.304J, 0.374J and 0.363J for the sequential, pure data parallelism and mixed parallelism implementations. The average static energy decline rate for all the 3 image sizes are: -19.56% for the pure data parallelism and -18.22% for the mixed parallelism implementation for the Emesh architecture as compared to the sequential implementation. The pure data and mixed implementations show an average decline rate of -23.38% and -19.48% respectively, against sequential implementation for all the 3 image sizes in ATAC. Comparison of the average values for all the 3 image sizes for ORNoC architecture indicates that there is -23.03% and -19.41% degradation in the static energy consumption values respectively, for pure data parallelism and mixed parallelism implementations against Emesh. From the results, the pure data parallelism implementation consumes more static energy as works on subdivided image blocks on all the stages which requires more energy which is data independent while the mixed implementation uses a mixture of parallelization schemes which has a sequential implementation stage that does not require splitting the image and hence, consumes lesser energy as compared to pure data implementation.

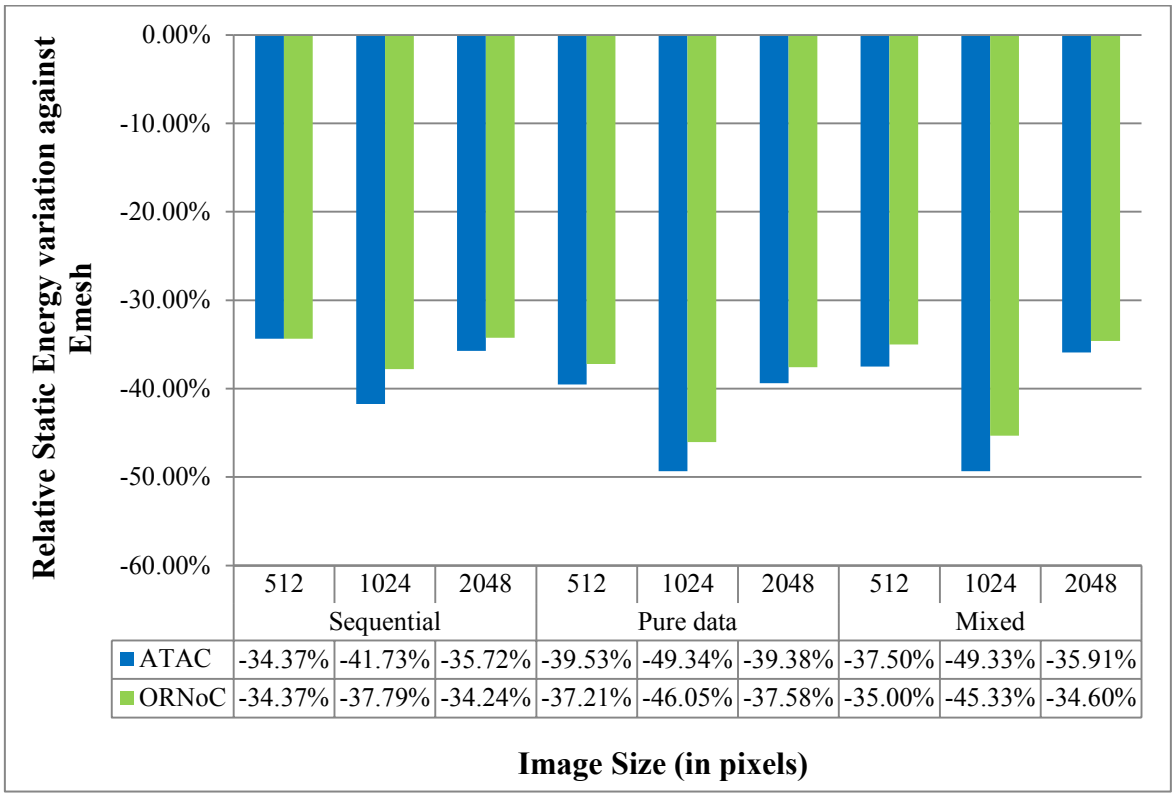
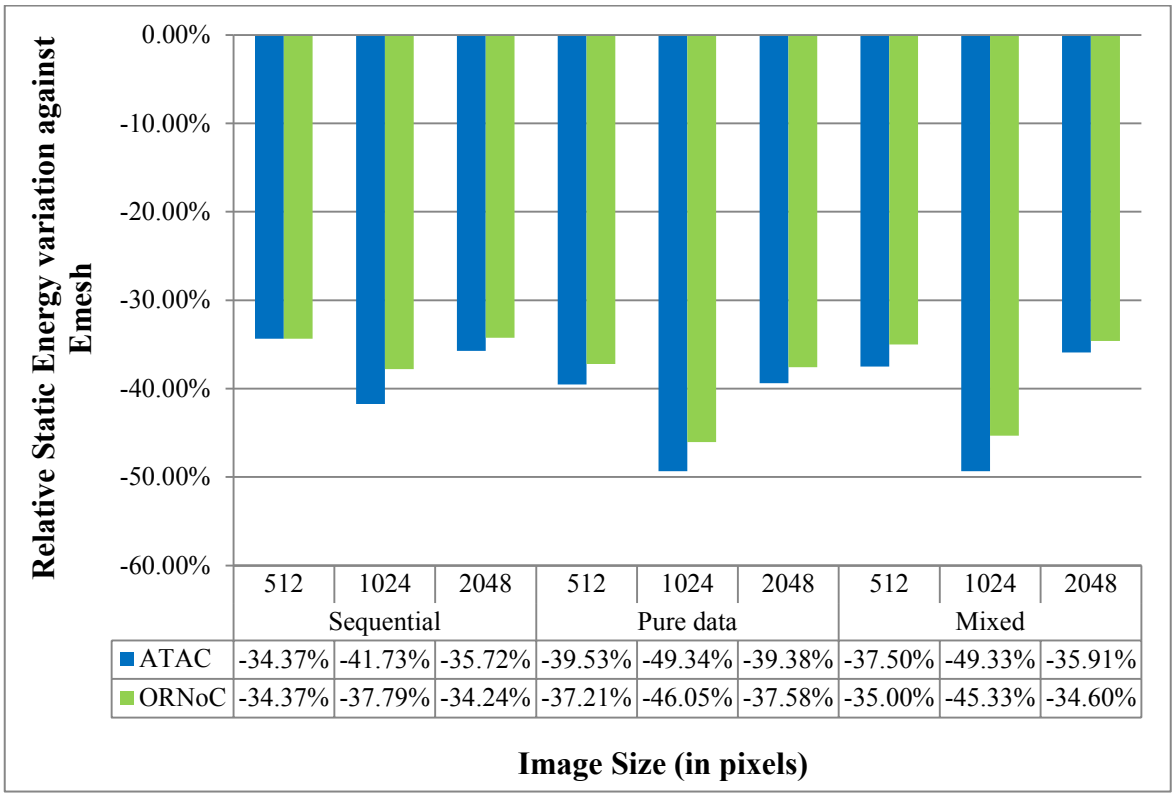


Figure 30: Relative Static Energy variation for Image Sizes NxN, N = {512, 1024, 2048}

The variation in static energy for the ONoC architectures against Emesh is displayed in the diagram in

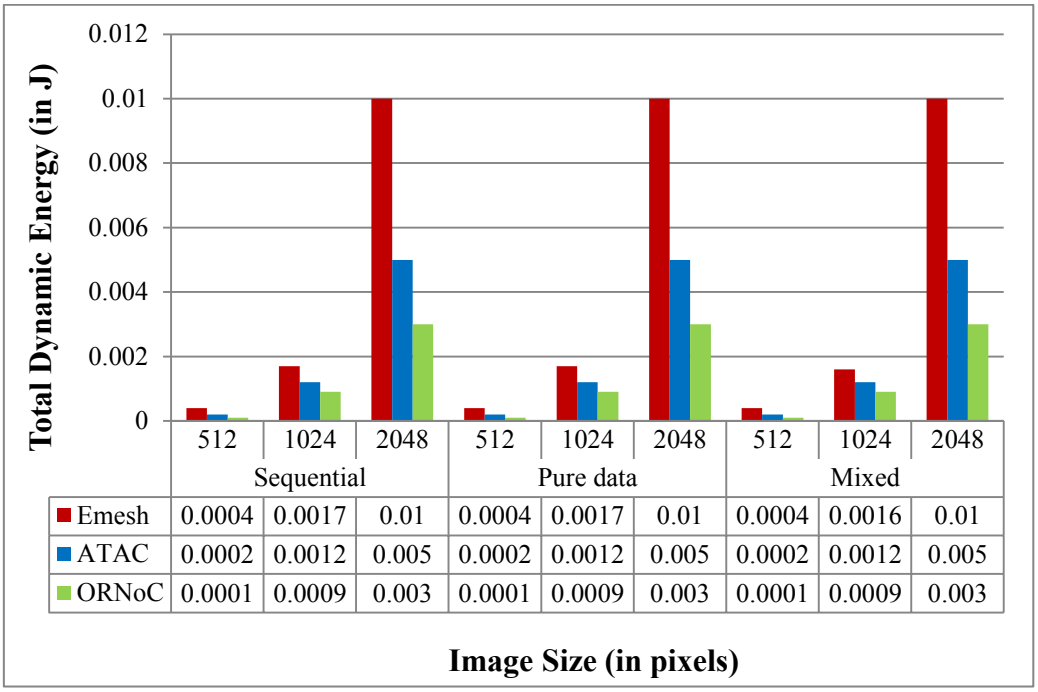


	512	1024	2048	512	1024	2048	512	1024	2048
	Sequential			Pure data			Mixed		
ATAC	-34.37%	-41.73%	-35.72%	-39.53%	-49.34%	-39.38%	-37.50%	-49.33%	-35.91%
ORNoC	-34.37%	-37.79%	-34.24%	-37.21%	-46.05%	-37.58%	-35.00%	-45.33%	-34.60%

Figure 30. The negative values indicate a decline in the performance relative to static energy values of Emesh. Static energy consumption shows a decline in performance with increase in the image sizes. The relative static energy variation values for ATAC range from -34.37% for 512 pixel square image to -35.72% for 2048 pixel square image in the case of sequential implementation, -39.53% to -39.38% in the case of 512 pixel square image and 2048 pixel square image respectively for pure data parallelism implementation and -37.50% to -35.91% in the case of mixed parallelism implementation for 512 pixel square image and 2048 pixel square image respectively. The values of relative static energy variation for ORNoC range from -34.37% for 512 pixel square image to -34.24% for 2048 pixel square image in the case of sequential implementation, -37.21% to -37.58% in the case of 512 pixel square image and 2048 pixel square image respectively for pure data parallelism implementation and -35% to -34.6% in the case of mixed parallelism implementation for 512 pixel square image and 2048 pixel square image respectively. The sequential, pure data parallelism and mixed

implementations have average variations of -37.27%, -42.75% and -40.91% respectively for ATAC for all the 3 image sizes. The ORNoC architecture has average variations of -35.47%, -40.28% and -38.31% for sequential, pure data parallelism and mixed parallelism implementation schemes respectively for all the 3 image sizes. Emesh architecture shows better static energy consumption values as compared to both the ONoC architectures as there are no optical components in Emesh to constitute the calculation for data independent energy. The optical components introduce energy consumption for laser, thermal and ring tuning etc. ORNoC shows lesser static energy consumption than ATAC as the number of wavelengths and waveguides used are lesser, which in turn reduces the number of laser sources and the losses due to thermal tuning and ring tuning [54].

5.1.4 Results for Dynamic Energy for varying Image Sizes



	512	1024	2048	512	1024	2048	512	1024	2048
Emesh	0.0004	0.0017	0.01	0.0004	0.0017	0.01	0.0004	0.0016	0.01
ATAC	0.0002	0.0012	0.005	0.0002	0.0012	0.005	0.0002	0.0012	0.005
ORNoC	0.0001	0.0009	0.003	0.0001	0.0009	0.003	0.0001	0.0009	0.003

Figure 31: Total Dynamic Energy for Image Sizes NxN, N = {512, 1024, 2048}

The increase in the image size causes an increase in the dynamic energy consumption values for all the architectures as the dynamic energy is data dependent. For the Emesh architecture, the dynamic energy consumption increases from 0.0004J when the image size is 512 x 512 pixels to 0.01J for 2048 x 2048 pixels in the case of all the three implementations. The ATAC architecture shows the dynamic energy consumption values from 0.0002J when the image size is 512 x 512 pixels to 0.005J for 2048 x 2048 pixels for all the implementations. For the ORNoC architecture, the dynamic energy values increase from 0.0001J when the image size is 512 x 512 pixels to 0.003J for 2048 x 2048 pixels in the case of all the three implementations. The dynamic energy consumption values for all the 3 images are observed to be an average of 0.004J for Emesh, 0.0021J for ATAC and 0.0013J for ORNoC architectures in case of all the three implementation schemes. The values of dynamic energy consumption are observed to be equal for each of the image size for each of the architectures irrespective of the implementation scheme.

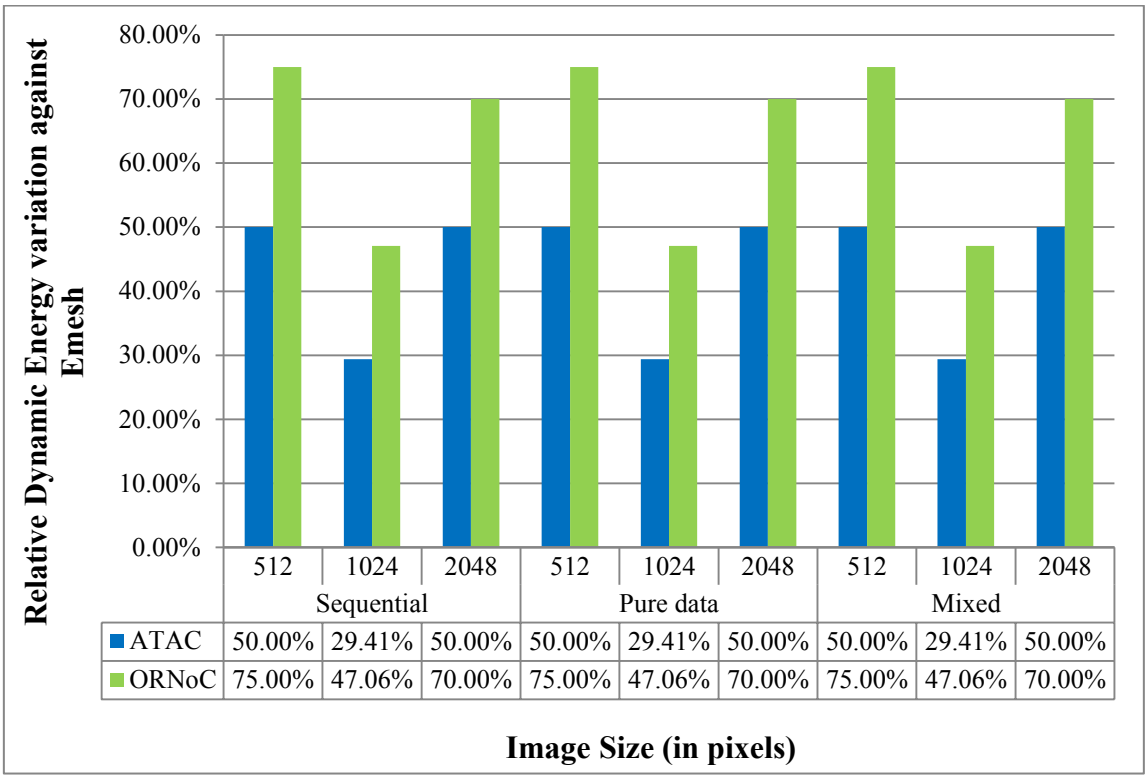


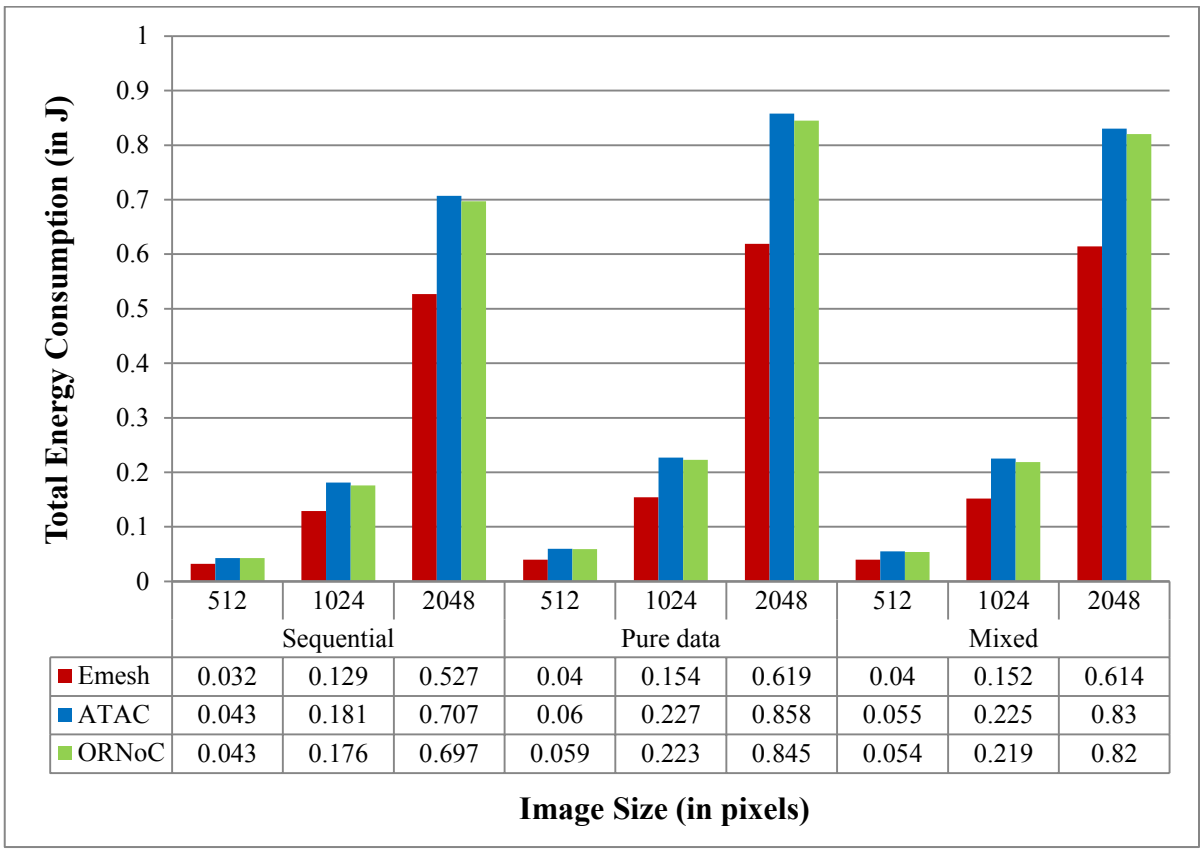
Figure 32: Relative Dynamic Energy variation for Image Sizes NxN, N = {512, 1024, 2048}

The variation in the dynamic energy against Emesh is plotted in Figure 32. The relative dynamic energy variation values for ATAC range from 50% for 512 pixel square image, reduces for 1024 pixel square image and goes back to 50% for 2048 pixel square image for all the implementation schemes . The relative latency variation values for ORNoC range from 75% for 512 pixel square image and reduces to 47.06% for 1024 pixel square image to 70% for 2048 pixel square image for all the 3 implementation schemes .

The average values for all the 3 images for the relative variation of dynamic energy consumption against Emesh for the same image size are 43.14% for ATAC and 64.02% for ORNoC architectures across all the three implementation schemes. Emesh architecture is observed to have higher dynamic energy consumption as compared to the ONoC architectures, because of larger number of electrical links [54]. Among the ONoC

architectures, ATAC has higher energy consumption than ORNoC as ORNoC reuses the wavelengths which causes a decrease in the number of receivers or modulators which constitute data dependent energy [54].

5.1.5 Results for Total Energy for varying Image Sizes



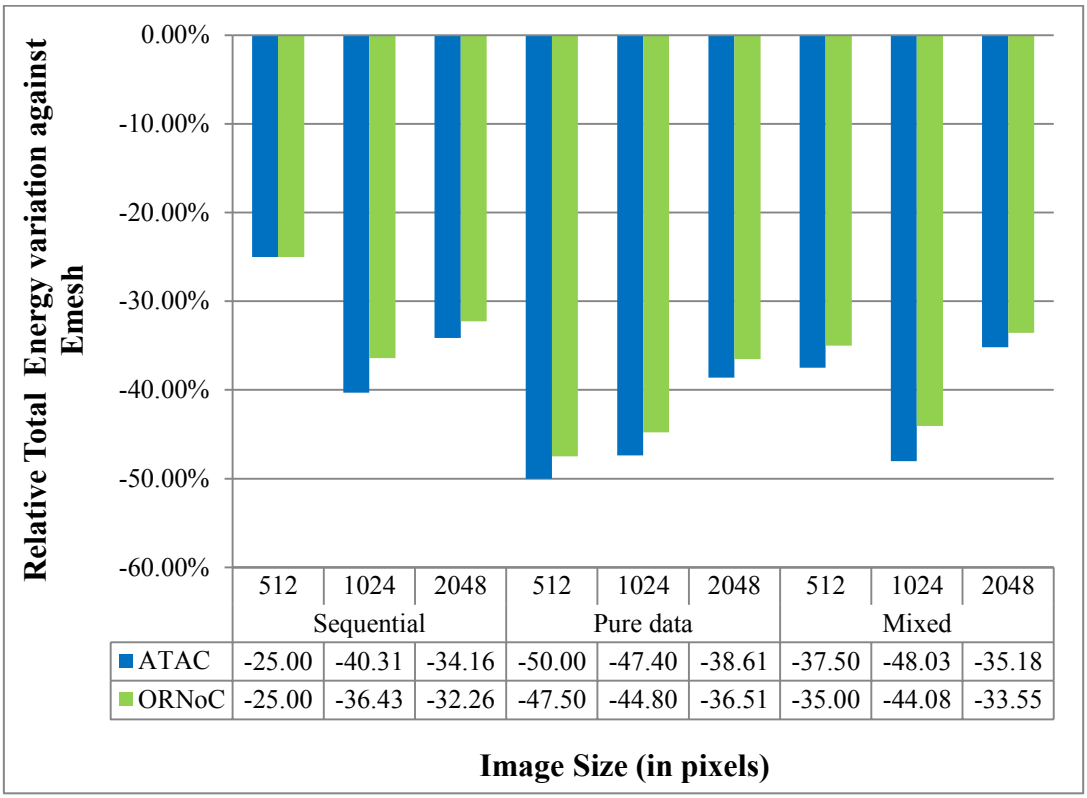
	512	1024	2048	512	1024	2048	512	1024	2048
	Sequential			Pure data			Mixed		
Emesh	0.032	0.129	0.527	0.04	0.154	0.619	0.04	0.152	0.614
ATAC	0.043	0.181	0.707	0.06	0.227	0.858	0.055	0.225	0.83
ORNoC	0.043	0.176	0.697	0.059	0.223	0.845	0.054	0.219	0.82

Figure 33: Total Energy Consumption for Image Sizes NxN, N = {512, 1024, 2048}

The total energy consumption gives the sum of both static and dynamic energy consumption. The predominant contributor towards the total energy consumption is static energy. The total energy consumption increases with increase in workload, similar to both of its constituent components, for all the architectures. For the Emesh architecture, the total energy consumption values increase from 0.032J when the image size is 512 x 512 pixels to 0.527J for 2048 x 2048 pixels in the case of sequential implementation, while it is between 0.04J

and 0.619J for pure data parallelism implementation for 512 pixel square image and 2048 pixel square image respectively and the values are between 0.04J for 512 pixel square image and 0.614J for the 2048 pixel square image for the mixed implementation. The ATAC architecture shows the total energy consumption values from 0.043J when the image size is 512 x 512 pixels to 0.707J for 2048 x 2048 pixels in the case of sequential implementation, while it is between 0.06J and 0.858J for pure data parallelism implementation for 512 pixel square image and 2048 pixel square image respectively and the values are between 0.055J for 512 pixel square image and 0.83J for the 2048 pixel square image for the mixed implementation. For the ORNoC architecture, the total energy consumption values increase from 0.043J when the image size is 512 x 512 pixels to 0.697J for 2048 x 2048 pixels in the case of sequential implementation, while it is between 0.059J and 0.845J for pure data parallelism implementation for 512 pixel square image and 2048 pixel square image respectively and the values are between 0.054J for 512 pixel square image and 0.82J for the 2048 pixel square image for the mixed implementation. The average values for all the 3 images in the case of total energy consumption are 0.229J for Emesh, 0.310J for ATAC and 0.305J for ORNoC in the case of sequential implementation. The average values for all the 3 images for pure data parallelism implementation are 0.271J, 0.382J and 0.376J for Emesh, ATAC and ORNoC architectures respectively. The average total energy consumption values for mixed parallelism implementation are 0.269J for Emesh, 0.37J for ATAC and 0.364J for ORNoC. The average variation for all the 3 images in the total energy consumption values against sequential implementation for pure data parallelism and mixed parallelism implementations are -18.34% and -17.47% respectively for Emesh, -23.23% and -19.35% respectively for ATAC and -23.28% and -19.34% respectively for the ORNoC architectures.

The values for Emesh are lower as compared to both ONoC architectures. ORNoC has the same values as compared to ATAC for all the workloads for except when the image size is 2048 pixels, where it is slightly lesser than ATAC. The values for sequential implementation is lesser than both parallelized implementations, as the parallelized versions have the task of creating and assigning tasks to the threads. The mixed parallelization scheme shows slightly better energy consumption than the pure data scheme highlighting the advantages of effective parallelization.



	512	1024	2048	512	1024	2048	512	1024	2048
	Sequential			Pure data			Mixed		
ATAC	-25.00	-40.31	-34.16	-50.00	-47.40	-38.61	-37.50	-48.03	-35.18
ORNoC	-25.00	-36.43	-32.26	-47.50	-44.80	-36.51	-35.00	-44.08	-33.55

Figure 34: Relative Total Energy variation for Image Sizes NxN, N = {512, 1024, 2048}

The relative variation in total energy for the ONoC architectures against Emesh is shown in the above plot. The relative total energy values for ATAC range from -25% for 512 pixel square image to -34.16% for 2048 pixel square image in the case of sequential implementation, -50% to -38.61% in the case of 512 pixel square image and 2048 pixel

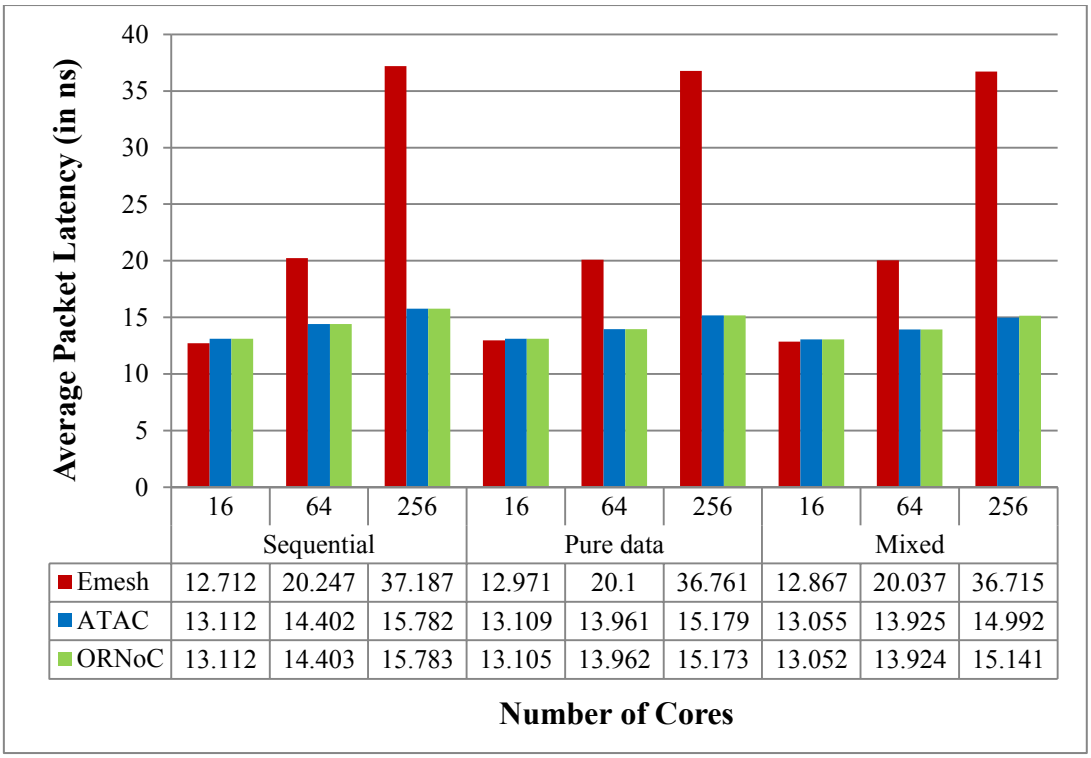
square image respectively for pure data parallelism implementation and -37.50% to -35.18% in the case of mixed parallelism implementation for 512 pixel square image and 2048 pixel square image respectively. The relative latency variation values for ORNoC range from -25% for 512 pixel square image to -32.26% for 2048 pixel square image in the case of sequential implementation, -47.5% to -36.51% in the case of 512 pixel square image and 2048 pixel square image respectively for pure data parallelism implementation and -35% to -33.55% in the case of mixed parallelism implementation for 512 pixel square image and 2048 pixel square image respectively. The average decline in the values are -33.16% for ATAC and -31.23% for ORNoC in the case of sequential implementation for all the 3 images, -45.34% for ATAC and -42.94% for ORNoC in the case of pure data parallelism implementation for all the 3 images, -40.24% for ATAC and -37.54% for ORNoC in the case of mixed parallelism implementation for all the 3 images. The trend followed is the same as that of static energy consumption due to similar justifications as mentioned in section 5.2.3.

5.2 Analysis with varying number of Cores

This set of experiments involves executing the application on different number of cores for each of the implementations. The experimental cases include core numbers of 16, 64 and 256. The only values for the number of cores which are considered for this experiment are even powers of two greater than or equal to 16, since the odd powers of two cannot form a proper square mesh for all the architectures. The experimental cases use the cluster size of 8 to allow a value permissible for all the cores and the application utilizes a square image of size 1024 pixels. The cluster size remains same, the values throughout the experiment, which

leads to more number of clusters for increased number of cores. . The image size is chosen as 1024 as it provides a moderate workload for all the different number of cores under consideration. The latency and energy outputs of the network are observed to determine the choice of optimal number of cores for the user. As per the observations from this experiment, the default value of core size for all the experiments have been kept as 64, as it highlights the increased delay for electrical architecture over the optical architectures and also offers adequate processing power for the image loads under consideration. The other parameters retain their default values as per Table 4. The specific configuration for this set of experiments is given in Appendix Table 2: Configurations for varying Number of Cores.

5.2.1 Results for Packet Latency for varying number of Cores



	16	64	256	16	64	256	16	64	256
	Sequential			Pure data			Mixed		
Emesh	12.712	20.247	37.187	12.971	20.1	36.761	12.867	20.037	36.715
ATAC	13.112	14.402	15.782	13.109	13.961	15.179	13.055	13.925	14.992
ORNoC	13.112	14.403	15.783	13.105	13.962	15.173	13.052	13.924	15.141

Figure 35: Average Packet Latency for Number of Cores N={16,64,256}

With the increase in number of cores, there is more number of clusters since the cluster size remains the same, which in turn increases the communication between clusters. At the same time, the larger number of cores with the same cluster size would increase the intra cluster communication due to increased electrical links and hence increase latency. Figure 35 shows the observations for average packet latency for all the cores. The average packet latency values increase with the increase in the number of cores for all the three architectures for the workload considered for the experiment. As observed from Figure 35, the increase in average packet latency is much higher for the Emesh than for either ONoC architectures. The sequential implementation for Emesh ranges from 12.712ns to 37.187ns for 16 cores to 256 cores respectively in the case of sequential implementation, from 12.971ns for 16 cores to 36.761ns for 256 cores in the case of pure data parallelism implementation and from 12.867ns for 16 cores to 36.715ns for 256 cores in the case of mixed parallelism implementation. ATAC shows the values from 13.112ns for 16 cores and 15.872ns for 256 cores in the case of sequential implementation, from 13.102ns to 15.719ns respectively for 16 cores and 26 cores respectively in the case of pure data parallelism and from 13.055ns for 16 cores to 14.992ns for 256 cores in the cased of mixed parallelism implementation. The values for ORNoC architecture range from 13.112ns for 16 cores to 15.783ns for 256 cores in the case of sequential implementation, from 13.105ns to 15.173ns for 16 and 256 cores respectively in the case of pure data parallelism implementation and 13.052ns from 16 cores to 15.141ns for 256 cores in the case of mixed parallelism implementation.

The average values are calculated for all the 3 core sizes. The average values for all the core sizes for Emesh architecture are 23.34ns, 23.38ns and 23.21ns respectively for the sequential, pure data parallelism and mixed parallelism implementations. The results for

ATAC show average values of 14.43ns, 14.08ns and 13.99ns for sequential, pure data parallelism and mixed data implementations. ORNoC results are different from ATAC only for the mixed parallelism implementation, which shows the average value of 14.03ns. This indicates a 0.6 % decline in the pure data parallelism performance and 0.5% increase in mixed implementation performance for the Emesh architecture as compared to the sequential implementation. The pure data and mixed implementations show an average improvement of 2.43% and 3.05% respectively, against sequential implementation in ATAC. Comparison of average values for ORNoC architecture indicate that there is 2.43% and 2.77% improvement in the latency values respectively, for pure data parallelism and mixed parallelism implementations against Emesh. Among the implementations, the mixed implementation gives the best latency values relative to the sequential implementation due to efficient use of parallelization strategies as compared to the pure data parallelism implementation. Also, we can see that having 16 cores total does not improve the performance, which has only two clusters and hence, there will not be sufficient traffic in the optical network for the ONoC architectures and the performance of the ONoC architectures will be minimal as compared to greater number of clusters.

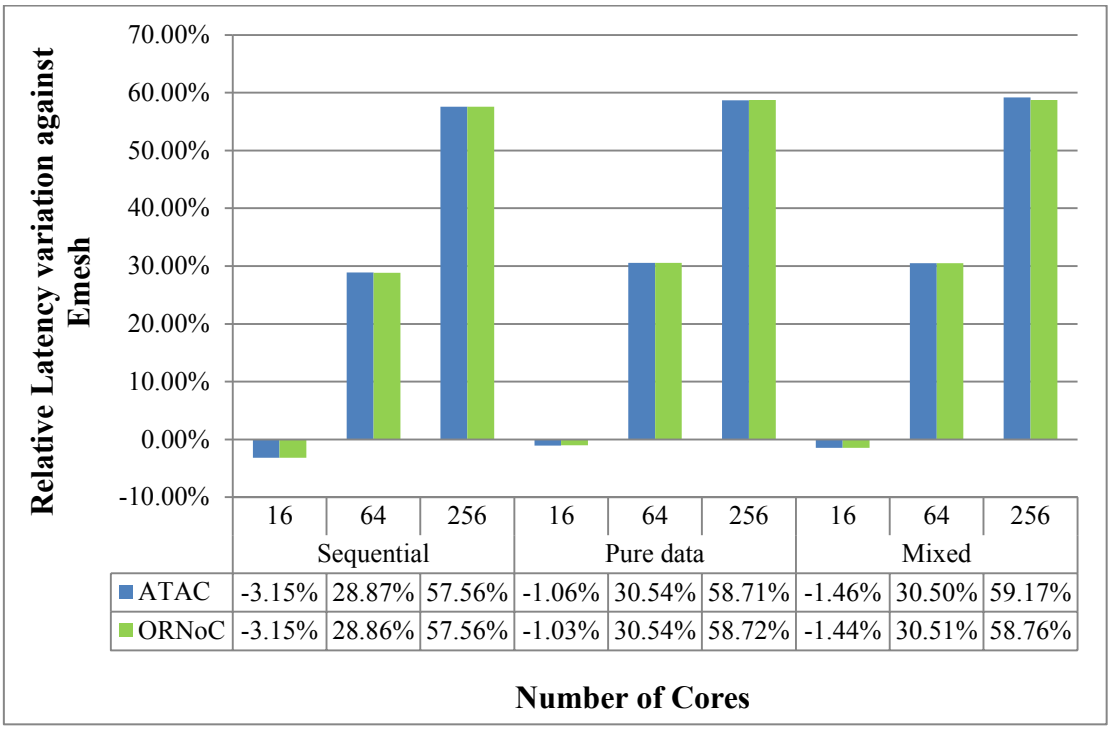


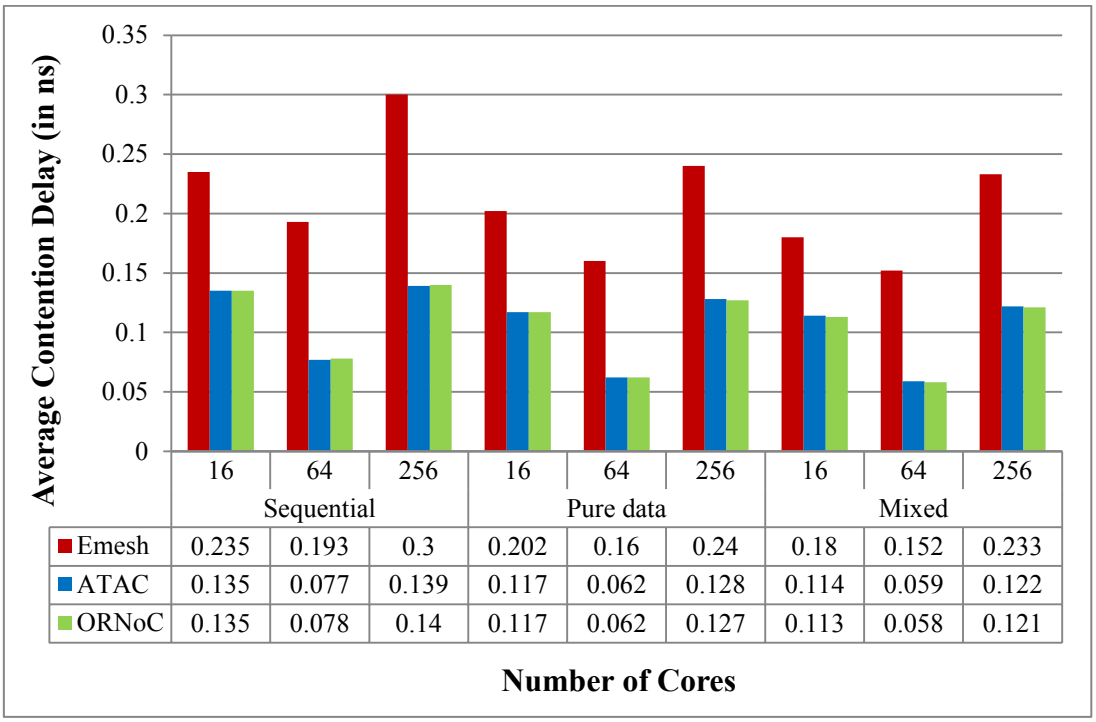
Figure 36: Relative Latency variation for Number of Cores N={16,64,256}

The relative total energy improvement goes from -3.15% for 16 cores to 57.56% for 256 cores in the case of sequential implementation, -1.06% for 16 cores to 58.71% for 256 cores in the case of pure data parallelism implementation and -1.46% for 16 cores and 59.17% for 256 cores in the case of mixed parallelism implementation for the ATAC architecture. ORNoC architecture shows relative total energy values of -3.15% for 16 cores to 57.56% for 256 cores in the case of sequential implementation, -1.03% for 16 cores to 58.72% for 256 cores in the case of pure data parallelism implementation and -1.44% for 16 cores to 58.76% for 256 cores in the case of mixed parallelism implementation.

The average values are calculated for all the three core sizes considered for the experiment. The values show an average increase of 27.76% for the sequential implementation and 29.40% for both pure data parallelism implementation and mixed implementation for ATAC against Emesh. Whereas, the average increase in ORNoC values against Emesh are observed

to be 27.76%, 29.41% and 29.28% for the sequential, pure data parallelism and mixed parallelism implementations. The increased packet latency for Emesh architecture when compared to the ONoC architectures can be attributed to the poor scalability and absence of optical network for efficient communication. Both the ONoC architectures show very similar variation in latency values.

5.2.2 Results for Contention Delay for varying number of Cores



	16	64	256	16	64	256	16	64	256
	Sequential			Pure data			Mixed		
Emesh	0.235	0.193	0.3	0.202	0.16	0.24	0.18	0.152	0.233
ATAC	0.135	0.077	0.139	0.117	0.062	0.128	0.114	0.059	0.122
ORNoC	0.135	0.078	0.14	0.117	0.062	0.127	0.113	0.058	0.121

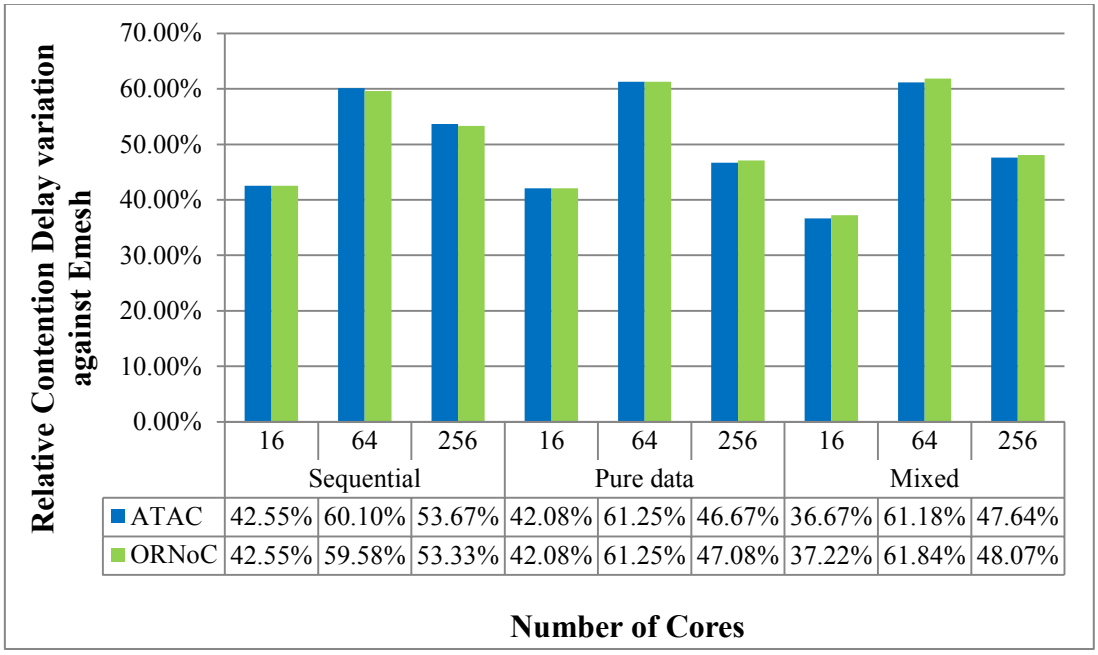
Figure 37: Average Contention Delay for Number of Cores N={16,64,256}

The contention delay is expected to increase with increase in core size, but according to the experimental settings, the cluster size is 8, which means that there are only two clusters available for communication when the core size is 16, which limits the communication in the optical network and hence, increases the contention delay. The optical network has sufficient communication for the considered workload, when the number of clusters is 8, in the case of

64 cores and hence, the improvement in values. For 256 cores, there is increased distribution of workload necessary as there is more number of cores involved while the cluster number remains the same which again increases the contention. The sequential implementation for Emesh ranges from 0.235ns to 0.3ns for 16 cores to 256 cores respectively in the case of sequential implementation, from 0.202ns for 16 cores to 0.24ns for 256 cores in the case of pure data parallelism implementation and from 0.18ns for 16 cores to 0.233ns for 256 cores in the case of mixed parallelism implementation. ATAC shows the values from 0.135ns for 16 cores and 0.139ns for 256 cores in the case of sequential implementation, from 0.117ns to 0.128ns respectively for 16 cores and 26 cores respectively in the case of pure data parallelism and from 0.114ns for 16 cores to 0.122ns for 256 cores in the cased of mixed parallelism implementation. The values for ORNoC architecture range from 0.135ns for 16 cores to 0.14ns for 256 cores in the case of sequential implementation, from 0.117ns to 0.127ns for 16 and 256 cores respectively in the case of pure data parallelism implementation and 0.113ns from 16 cores to 0.121ns in the case of mixed parallelism implementation.

The average values are calculated for all the 3 core sizes. The average values of contention delay for sequential implementation are 0.243ns, 0.117ns and 0.118ns for Emesh, ATAC and ORNoC architectures respectively. Pure data parallelism implementation gives average values of 0.201ns for Emesh, 0.102ns for both ATAC and ORNoC architectures. The average values for mixed implementation are 0.188ns, 0.098ns and 0.097ns respectively for Emesh, ATAC and ORNoC architectures. The average variation between the pure data parallelism and mixed parallelism against sequential implementation are 17.28% and 22.63% for Emesh, 12.82% and 16.24% for ATAC and 13.56% and 17.8% for ORNoC. The

improvement in contention delay values against sequential implementation for mixed parallelism implementation is observed to be higher than that of pure data parallelism implementation due to the more effective parallelization procedure used in the mixed parallelism implementation.



	Sequential			Pure data			Mixed		
	16	64	256	16	64	256	16	64	256
ATAC	42.55%	60.10%	53.67%	42.08%	61.25%	46.67%	36.67%	61.18%	47.64%
ORNoC	42.55%	59.58%	53.33%	42.08%	61.25%	47.08%	37.22%	61.84%	48.07%

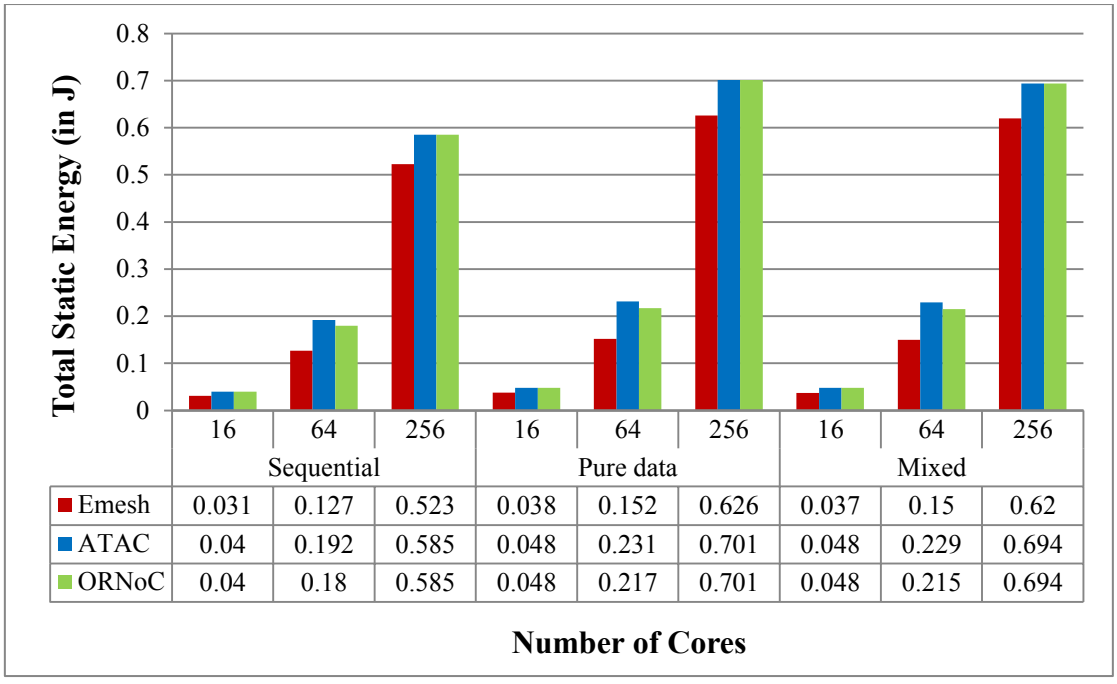
Figure 38: Relative Contention Delay for Number of Cores N={16,64,256}

The improvement in contention delay values for ONoC architectures show an initial increase from 16 to 64 cores and reduces thereafter for 256 cores. The relative total energy improvement goes from 42.55% for 16 cores to 53.67% for 256 cores in the case of sequential implementation, 42.08% for 16 cores to 46.67% for 256 cores in the case of pure data parallelism implementation and 36.67% for 16 cores and 47.64% for 256 cores in the case of mixed parallelism implementation for the ATAC architecture. ORNoC architecture shows relative total energy values of 42.55% for 16 cores to 53.33% for 256 cores in the case of sequential implementation, 42.08% for 16 cores to 47.08% for 256 cores in the case

of pure data parallelism implementation and 37.22% for 16 cores to 48.07% for 256 cores in the case of mixed parallelism implementation.

The average values for the relative variation is calculated for all the core sizes considered. The contention delay values of ATAC show an average increase of 52.11% for the sequential, 50% for pure data parallelism and 48.5% for the mixed implementations against Emesh. Whereas, the average increase in ORNoC values against Emesh are observed to be 51.82%, 50.14% and 49.04% for the sequential, pure data parallelism and mixed parallelism implementations. The ONoC architectures show much better contention delay than Emesh for all the implementations as they have an optical network for effective communication. ORNoC and ATAC have contention delay values which are very close to each other.

5.2.3 Results for Static Energy for varying number of Cores



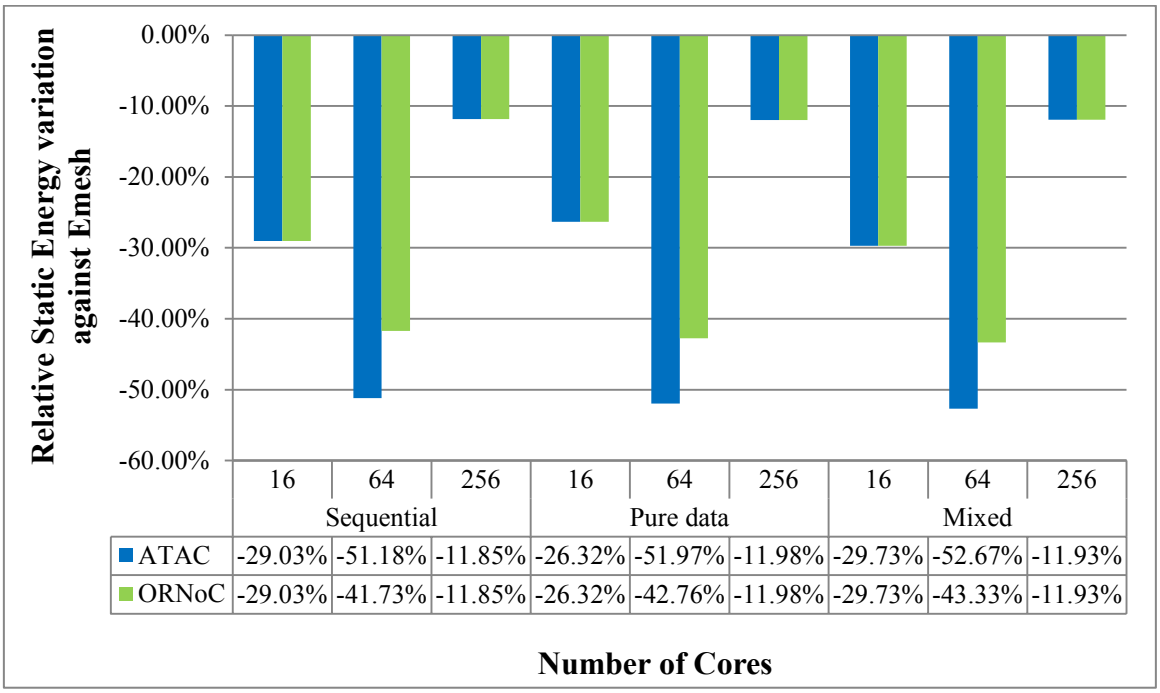
	16	64	256	16	64	256	16	64	256
	Sequential			Pure data			Mixed		
Emesh	0.031	0.127	0.523	0.038	0.152	0.626	0.037	0.15	0.62
ATAC	0.04	0.192	0.585	0.048	0.231	0.701	0.048	0.229	0.694
ORNoC	0.04	0.18	0.585	0.048	0.217	0.701	0.048	0.215	0.694

Figure 39: Total Static Energy for Number of Cores N={16,64,256}

Increase in the values for the number of cores, shows an increase in the static energy consumption values. As the number of clusters increases with the increase in the number of cores, the number of optical components that contribute to static energy consumption increases. The static energy consumption for Emesh ranges from 0.031J to 0.523J for 16 cores to 256 cores respectively in the case of sequential implementation, from 0.038J for 16 cores to 0.626J for 256 cores in the case of pure data parallelism implementation and from 0.037J for 16 cores to 0.62J for 256 cores in the case of mixed parallelism implementation. Both ATAC and ORNoC architectures show the values from 0.04J for 16 cores and 0.585J for 256 cores in the case of sequential implementation, from 0.048J to 0.701J respectively for 16 cores and 26 cores respectively in the case of pure data parallelism and from 0.048J for 16 cores to 0.694J for 256 cores in the cased of mixed parallelism implementation. The values for ORNoC architecture range from 13.112ns for 16 cores to 15.783ns for 256 cores in the case of sequential implementation, from 13.105ns to 15.173ns for 16 and 256 cores respectively in the case of pure data parallelism implementation and 13.052ns from 16 cores to 15.141ns in the case of mixed parallelism implementation.

The average values are calculated for all the 3 core sizes. The average values of static energy consumption for sequential implementation are 0.227J, 0.273J and 0.268J for Emesh, ATAC and ORNoC architectures respectively. Pure data parallelism implementation shows average values of 0.272J for Emesh, 0.327J for ATAC and 0.322J for ORNoC architectures. The average values for mixed implementation are 0.269J, 0.324J and 0.319J respectively for Emesh, ATAC and ORNoC architectures. The average variation in static energy values between the pure data parallelism and mixed parallelism against sequential implementation are -19.82% and -18.50% for Emesh, -20.22% and -19.12% for ATAC and -20.15% and -

19.03% for ORNoC respectively, where the negative variation indicates the degeneration or increased energy consumption. Sequential implementation shows minimal static energy consumption values which can be attributed to the lesser number of packets in the network which in turn reduces the number of active data independent components. Both the parallelized implementations have almost similar number of packets in the network, with the pure data parallelism implementation having slightly lesser number of packets than the mixed parallelism implementation.



	16	64	256	16	64	256	16	64	256
	Sequential			Pure data			Mixed		
ATAC	-29.03%	-51.18%	-11.85%	-26.32%	-51.97%	-11.98%	-29.73%	-52.67%	-11.93%
ORNoC	-29.03%	-41.73%	-11.85%	-26.32%	-42.76%	-11.98%	-29.73%	-43.33%	-11.93%

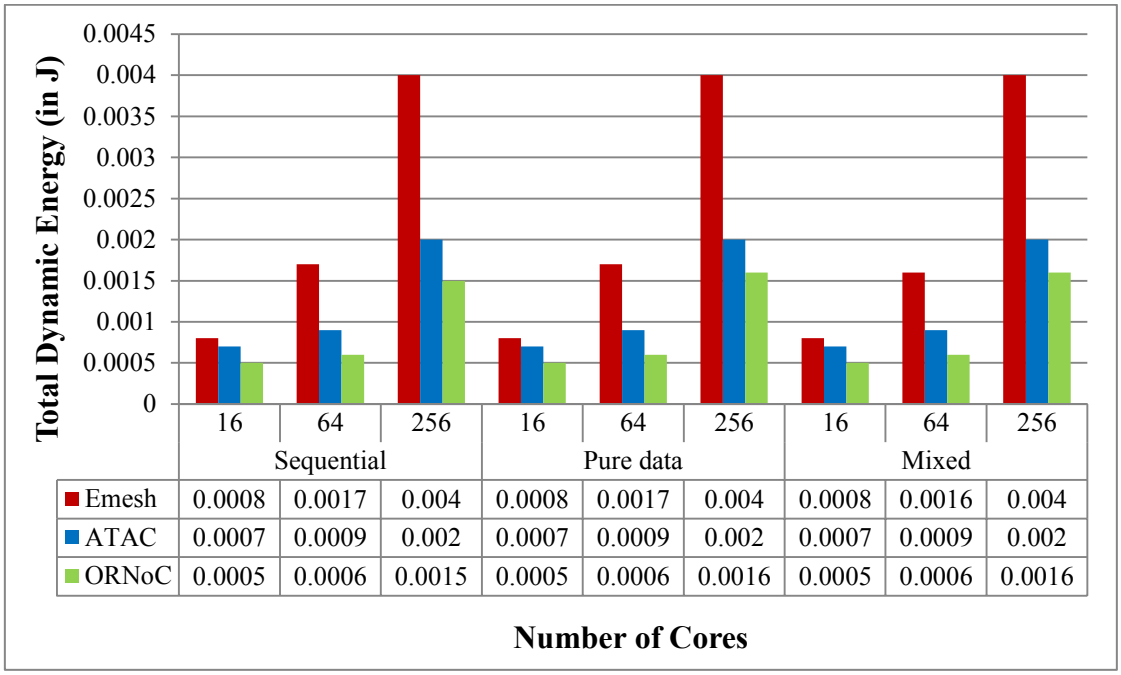
Figure 40: Relative Static Energy variation for Number of Cores N={16,64,256}

The relative total static energy improvement goes from -29.03% for 16 cores to -11.85% for 256 cores in the case of sequential implementation, -26.32% for 16 cores to -11.98% for 256 cores in the case of pure data parallelism implementation and -29.73% for 16 cores and -11.93% for 256 cores in the case of mixed parallelism implementation for the ATAC architecture. ORNoC architecture shows relative total energy values of -29.03% for 16 cores to -11.85% for 256 cores in the case of sequential implementation, -26.32% for 16 cores to -

11.98% for 256 cores in the case of pure data parallelism implementation and -29.37% for 16 cores to -11.93% for 256 cores in the case of mixed parallelism implementation.

The average values for the relative variation is calculated for all the core sizes considered. The average variation in static energy with respect to Emesh is -30.69% and -27.54% for ATAC and ORNoC respectively in the case of sequential implementation. The ATAC architecture shows -30.09% and ORNoC shows -27.02% decline rate for pure data implementation. The mixed parallelism implementation shows -31.44% decrease rate for ATAC and -28.33% for ORNoC compared to the Emesh architecture. The relative static energy deteriorates initially as both Emesh and the ONoC architectures have more communication in the electrical network, and it increases when the number of cores is 64, and the difference decreases again as the number of cores increases as the distribution of data among larger cores causes the Emesh static energy consumption values to move closer to those of the ONoC architectures. The static energy consumption is the least for Emesh network from the observations as compared to ATAC and ORNoC due to the absence of optical components which contribute towards static energy [54]. ORNoC shows lesser static energy consumption than ATAC as the number of wavelengths and waveguides used are lesser, which in turn reduces the number of laser sources and the losses due to thermal tuning and ring tuning [54].

5.2.4 Results for Dynamic Energy for varying number of Cores



	16	64	256	16	64	256	16	64	256
	Sequential			Pure data			Mixed		
Emesh	0.0008	0.0017	0.004	0.0008	0.0017	0.004	0.0008	0.0016	0.004
ATAC	0.0007	0.0009	0.002	0.0007	0.0009	0.002	0.0007	0.0009	0.002
ORNoC	0.0005	0.0006	0.0015	0.0005	0.0006	0.0016	0.0005	0.0006	0.0016

Figure 41: Total Dynamic Energy for Number of Cores N = {16, 64, 256}

The total dynamic energy consumption increases with increasing number of cores for all the architectures. The sequential implementation for Emesh ranges from 0.0008J to 0.004J for 16 cores to 256 cores respectively in the case of all the three implementations. ATAC shows the values from 0.0007J for 16 cores to 0.002J for 256 cores in the case of all the implementations. The values for ORNoC architecture range from 0.0005J for 16 cores to 0.0015J for 256 cores in the case of sequential implementation, from 0.0005J to 0.0016J for 16 and 256 cores respectively in the case of pure data parallelism implementation and 0.0005J from 16 cores to 0.0016J in the case of mixed parallelism implementation.

The average values are calculated considering all the three core sizes. The average values for Emesh architectures are 0.0022J for both sequential implementation and pure data parallelism implementation, while it is 0.0021J for mixed parallelism implementation. The

average values are 0.0012J for ATAC and 0.0009J for ORNoC for both pure data parallelism and mixed parallelism implementations. The values are seen to be uniform for all the different implementation schemes for each core sizes in Figure 41. The dynamic energy contributes only a very minor fraction of the total energy consumption. All the parallelization schemes have almost uniform increase with almost same values of dynamic energy consumption. This matches the expectations as the workload is same for all the schemes with only changes in the network due to increase in the number of cores.

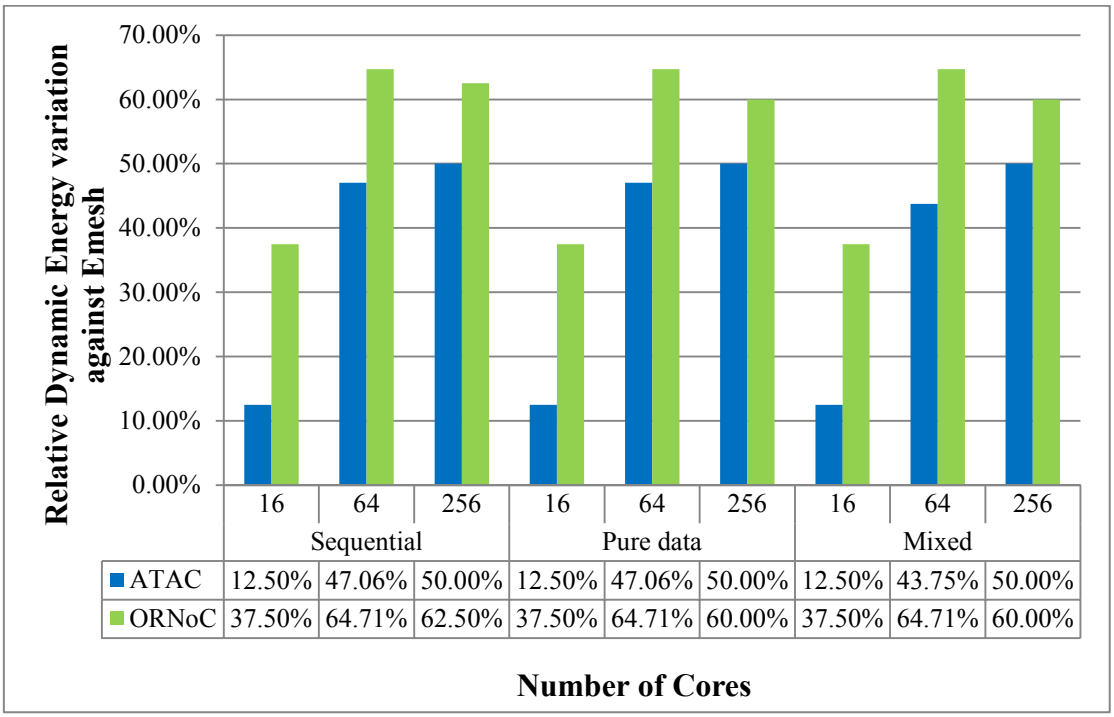
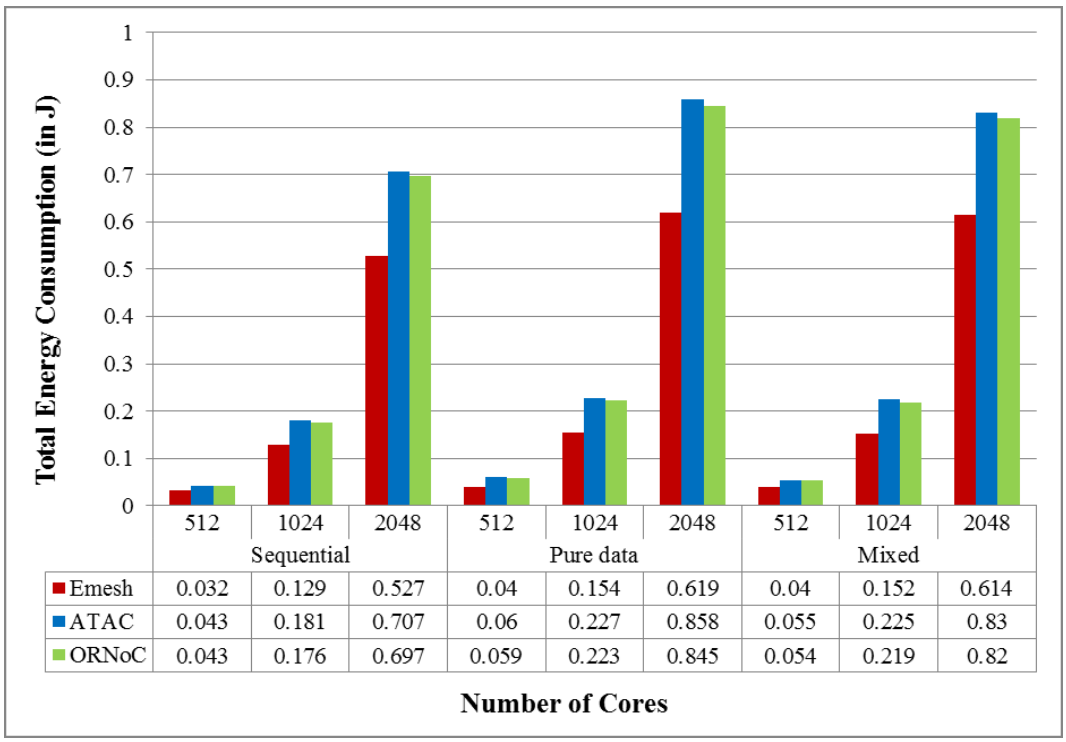


Figure 42: Relative Dynamic Energy variation for Number of Cores N={16,64,256}

The relative dynamic energy improvement goes from 12.50% for 16 cores to 50% for 256 cores for all the 3 implementations in the case of ATAC architecture. ORNoC architecture shows relative total energy values of 37.5% for 16 cores to 62.5% for 256 cores in the case of sequential implementation, 37.5% for 16 cores to 60% for 256 cores for both pure data parallelism implementation and mixed parallelism implementation.

The average values for the relative variation is calculated for all the core sizes considered. The average variation in dynamic energy with respect to Emesh is 36.52% and 54.92% for ATAC and ORNoC respectively in the case of sequential implementation. The ATAC architecture shows 36.52% and ORNoC shows 54.07% improvement rate for pure data implementation. The mixed parallelism implementation shows 35.42% improvement rate for ATAC and 54.07% for ORNoC compared to the Emesh architecture. The dynamic energy consumption is the highest for Emesh architecture while it is the lowest for ORNoC. Emesh architecture is observed to have slightly higher dynamic energy consumption as compared to the ONoC architectures, because of larger number of electrical links [54]. Among the ONoC architectures, ATAC has higher energy consumption than ORNoC as ORNoC reuses the wavelengths which causes a decrease in the number of receivers or modulators which constitute data dependent energy [54].

5.2.5 Results for Total Energy for varying number of Cores



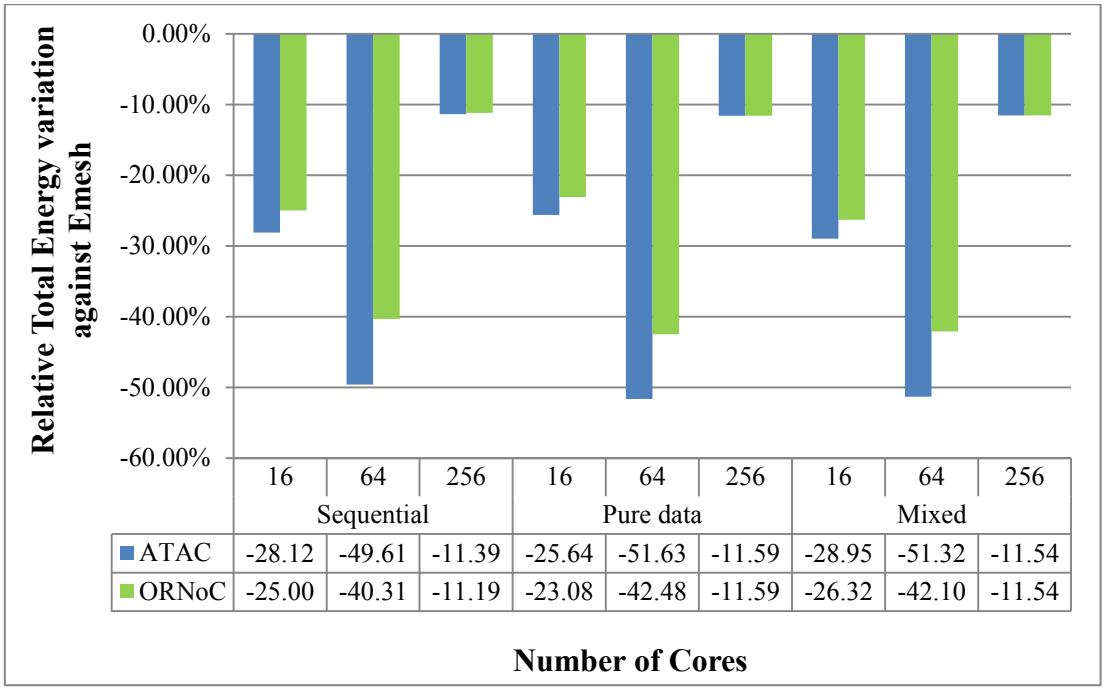
	512	1024	2048	512	1024	2048	512	1024	2048
	Sequential			Pure data			Mixed		
Emesh	0.032	0.129	0.527	0.04	0.154	0.619	0.04	0.152	0.614
ATAC	0.043	0.181	0.707	0.06	0.227	0.858	0.055	0.225	0.83
ORNoC	0.043	0.176	0.697	0.059	0.223	0.845	0.054	0.219	0.82

Figure 43: Total Energy Consumption for Number of Cores N = {16,64,256}

The total energy consumption gives the sum of dynamic and static energy consumption for different cores. Dynamic energy consumption contributes a minor fraction of the total energy consumption, while the major portion is contributed by static energy consumption and hence, total energy follows the same trend as that of static energy consumption. The sequential implementation for Emesh ranges from 0.032J to 0.527ns for 16 cores to 256 cores respectively in the case of sequential implementation, from 0.04J for 16 cores to 0.619J for 256 cores in the case of pure data parallelism implementation and from 0.04J for 16 cores to 0.614J for 256 cores in the case of mixed parallelism implementation. ATAC shows the values from 0.043J for 16 cores and 0.707J for 256 cores in the case of sequential implementation, from 0.06J to 0.858J respectively for 16 cores and 26 cores respectively in

the case of pure data parallelism and from 0.055J for 16 cores to 0.83J for 256 cores in the cased of mixed parallelism implementation. The values for ORNoC architecture range from 0.043J for 16 cores to 0.697J for 256 cores in the case of sequential implementation, from 0.059J to 0.845J for 16 and 256 cores respectively in the case of pure data parallelism implementation and 0.054J from 16 cores to 0.82J in the case of mixed parallelism implementation.

The average values are calculated for all the 3 core sizes. The average values are 0.229J, 0.310J and 0.305J for the Emesh, ATAC and ORNoC architectures respectively for the sequential implementation. Emesh architecture has 0.271J, ATAC architecture has 0.382J and ORNoC architecture has 0.376J as the average values for pure data parallelism implementation. Mixed parallelism implementation gives average values of 0.269J, 0.37J and 0.364J respectively for Emesh, ATAC and ORNoC architectures for total energy consumption.



	16	64	256	16	64	256	16	64	256
	Sequential			Pure data			Mixed		
ATAC	-28.12	-49.61	-11.39	-25.64	-51.63	-11.59	-28.95	-51.32	-11.54
ORNoC	-25.00	-40.31	-11.19	-23.08	-42.48	-11.59	-26.32	-42.10	-11.54

Figure 44: Relative Total Energy variation for Number of Cores N = {16, 64, 256}

The relative total energy improvement goes from -28.12% for 16 cores to -11.39% for 256 cores in the case of sequential implementation, -25.64% for 16 cores to -11.59% for 256 cores in the case of pure data parallelism implementation and -28.95% for 16 cores and -11.54% for 256 cores in the case of mixed parallelism implementation for the ATAC architecture. ORNoC architecture shows relative total energy values of -25% for 16 cores to -11.19% for 256 cores in the case of sequential implementation, -23.08% for 16 cores to -11.59% for 256 cores in the case of pure data parallelism implementation and -26.32% for 16 cores to -11.54% for 256 cores in the case of mixed parallelism implementation.

The average values for the relative variation is calculated for all the core sizes considered. The relative variation graph shows the degradation in energy consumption with respect to Emesh architecture. The sequential implementation shows average degeneration rate of -29.71% for ATAC and -25.5% for ORNoC relative to the Emesh architecture. The average degeneration rate for pure data parallelism implementation is observed to be -29.62% for ATAC and -25.72% for ORNoC against Emesh. The mixed parallelism implementation displays an average decline rate of -30.62% and -26.65% respectively, for ATAC and ORNoC architectures against Emesh.

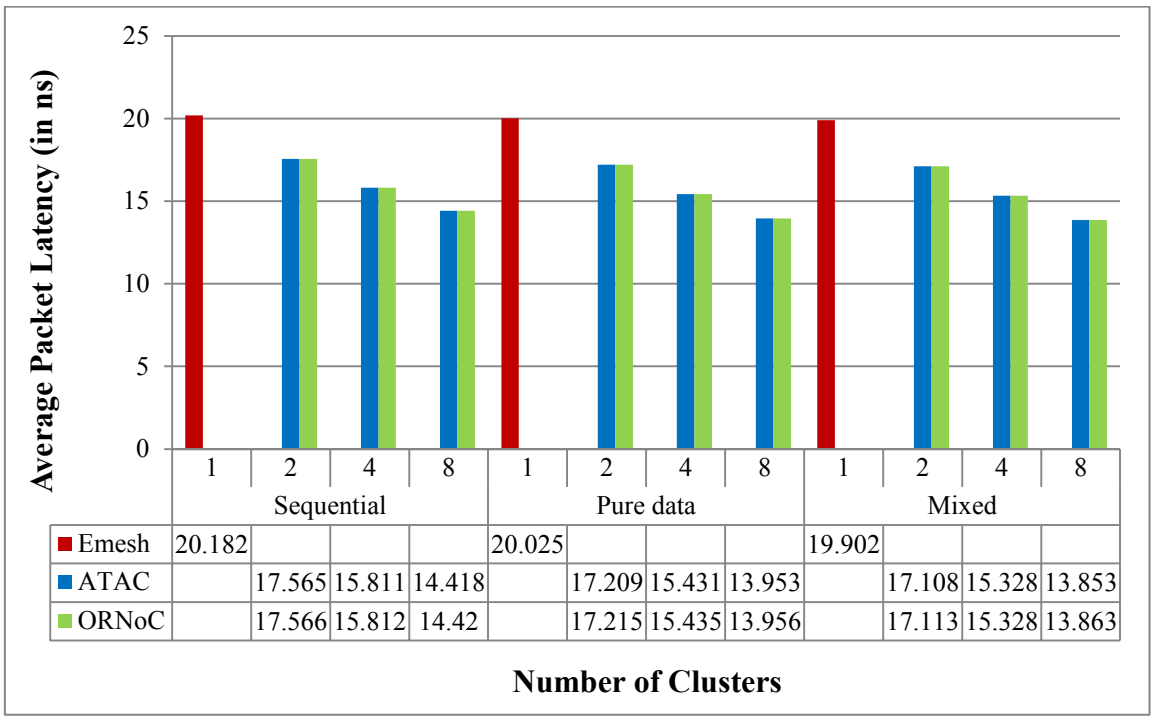
5.3 Analysis with varying number of Clusters

The cluster size indicates the number of cores that belong to a cluster and it controls the amount of traffic through the optical network. The cluster based routing is used for this set of experiments and hence, the intra cluster communication is through electrical networks and hence, larger cluster size for the same number of cores would mean lesser traffic in the optical network as there is lesser number of clusters.

The Emesh network architecture contains a square grid which provides electrical connectivity between the cores. For both the ONoC architectures, the intra cluster communication uses electrical signals while the inter cluster communication enables the usage of optical signals as the default routing strategy is cluster based routing. Hence, the Emesh network can be approximated to have only one cluster with cluster size is equal to the number of cores involved in the experiment and there is only a single configuration available for comparison.

The test cases include the number of clusters equal to 2 ,4 and 8. The experiment uses 64 cores and a 1024 pixel square image. The other permissible values for the number of cores are 16 and 256. While the former will have increased workload for the cores, the latter will have very little workload for all the cores. The image size is chosen as 1024 as it provides a moderate workload for all the different number of cores under consideration. The rest of the configurations are kept as default per Table 4. The detailed configurations for this set of experiments are given in **Error! Reference source not found.**. The inter cluster communication for larger images are bound to be more for ONoC architectures. After evaluating the results, it is seen that for this configuration, the ONoC architectures outperform Emesh in the delay outputs. as compared to other images of the set. The energy components are greater for ONoC architectures as static energy depends on optical components like laser power, thermal tuning etc.

5.3.1 Results for Packet Latency for varying number of Clusters



	Sequential				Pure data				Mixed			
	1	2	4	8	1	2	4	8	1	2	4	8
Emesh	20.182				20.025				19.902			
ATAC		17.565	15.811	14.418		17.209	15.431	13.953		17.108	15.328	13.853
ORNoC		17.566	15.812	14.42		17.215	15.435	13.956		17.113	15.328	13.863

Figure 45: Average Packet Latency for the Number of Clusters N = {1,2,4,8}

Figure 45 shows the average packet latency variation for the different cluster numbers. This figure is different form the others as Emesh has only one set of values for all the implementations while the ONoCs have all 3 sets of values. This remains the same for all the figures in this set of experiments. Average latency decreases with increase in cluster number for all the parallelization schemes for ATAC and ORNoC architectures. As the number of clusters increases, the communication in the optical network increases and hence, it accounts for the decrease in the latency values. The latency values are observed to be the best for mixed parallelization scheme, followed by pure data parallelization and then sequential scheme. The values for packet latency are 20.182ns for sequential, 0.116ns for pure data parallelism and 0.101ns for mixed parallelism implementation in the case of Emesh architecture. The ATAC architecture shows packet latency values ranging from 17.565ns for

2 clusters to 14.418ns for 8 clusters in the case of sequential implementation, 17.209ns for 2 clusters to 13.953ns for 8 clusters for pure data parallelism implementation, 17.108ns for 2 clusters to 13.853ns for 8 clusters in the case of mixed parallelism implementation. The packet latency values for the ORNoC architecture are from 17.566ns for 2 clusters to 14.42ns for 8 clusters in the case of sequential implementation, 17.215ns for 2 clusters to 13.956ns for 8 clusters for pure data parallelism implementation, 17.113ns for 2 clusters to 13.863ns for 8 clusters in the case of mixed parallelism implementation. The average values for the packet latency is calculated for all the cluster sizes considered. The average values of packet latency for sequential implementation are 20.182ns, 15.931ns and 15.933ns for Emesh, ATAC and ORNoC architectures respectively. Pure data parallelism implementation gives average values of 20.025ns for Emesh, 15.531ns for ATAC and 15.535ns for ORNoC architectures. The average values for mixed implementation are 19.902ns, 15.43ns and 15.435ns respectively for Emesh, ATAC and ORNoC architectures. The average variation between the pure data parallelism and mixed parallelism against sequential implementation are 0.77% and 1.39% for Emesh, 2.51% and 3.14% for ATAC and 2.5% and 3.12% for ORNoC respectively.

	2	4	8	2	4	8	2	4	8
	Sequential			Pure data			Mixed		
ATAC	12.97%	21.66%	28.56%	14.06%	22.94%	30.32%	14.04%	22.98%	30.39%
ORNoC	12.96%	21.65%	28.55%	14.03%	22.92%	30.31%	14.01%	22.98%	30.34%

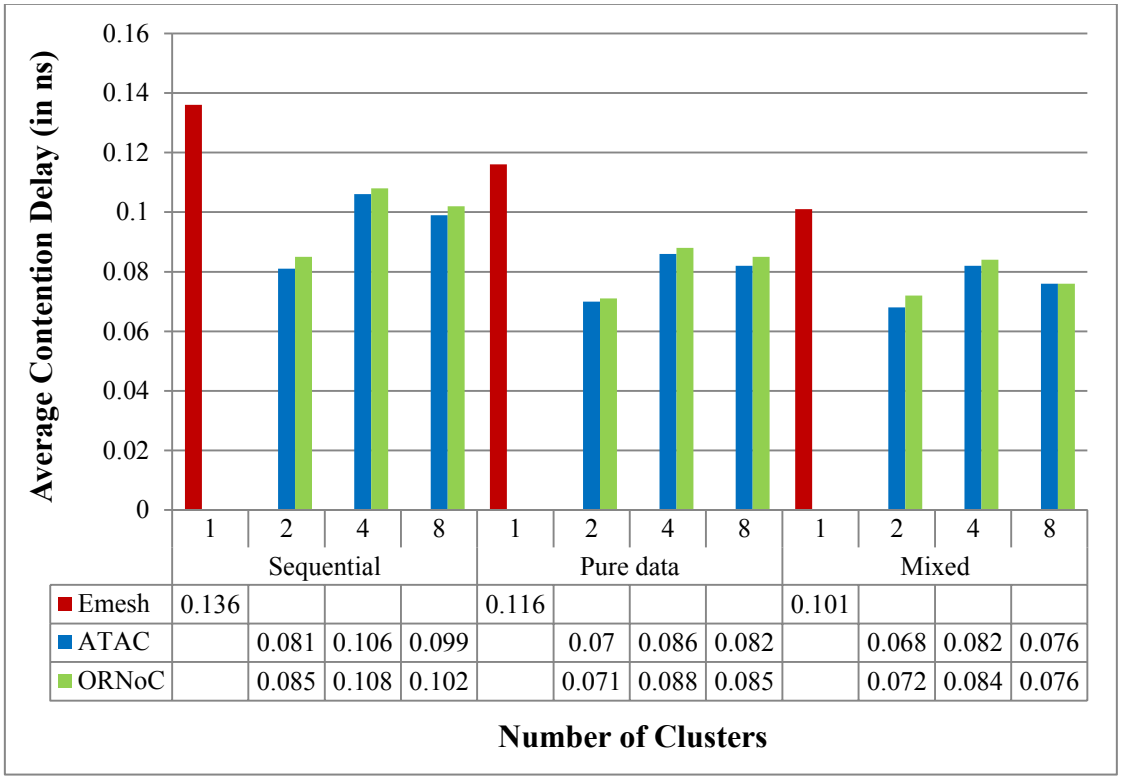
Figure 46: Relative Latency variation for Cluster Numbers N = {1,2,4,8}

The relative variation graph shows the improvement in packet latency for both ONoC architectures over Emesh. As expected, the improvement in the values increases with increasing number of clusters for both the ONoC architectures for all the implementation schemes. It is because the inter cluster communication uses the optical network, which increases with increasing number of clusters.

The relative contention delay improvement goes from 12.97% for 2 clusters to 28.56% for 8 clusters in the case of sequential implementation, 14.06% for 2 clusters to 30.32% for 8 lusters in the case of pure data parallelism implementation and 14.04% for 2 clusters and 30.39% for 8 clusters in the case of mixed parallelism implementation for the ATAC architecture. ORNoC architecture shows relative contention delay values of 12.96% for 2 clusters to 28.55% for 8 clusters in the case of sequential implementation, 14.03% for 2

clusters to 30.31% for 8 clusters in the case of pure data parallelism implementation and 14.01% for 2 clusters to 30.34% for 8 clusters in the case of mixed parallelism implementation. The average values for the relative variation is calculated for all the cluster sizes considered. The average increase in relative latency values are 21.06% for the sequential implementation and 22.44% for pure data parallelism implementation and 22.47% for mixed implementation for ATAC against Emesh. Whereas, the average increase in ORNoC values against Emesh are observed to be 21.05%, 22.42% and 22.44% for the sequential, pure data parallelism and mixed parallelism implementations. The latency values are greater for Emesh compared to the ONoC architectures for all three parallelization schemes because of the absence of optical network to efficiently handle network traffic.

5.3.2 Results for Contention Delay for varying number of Clusters



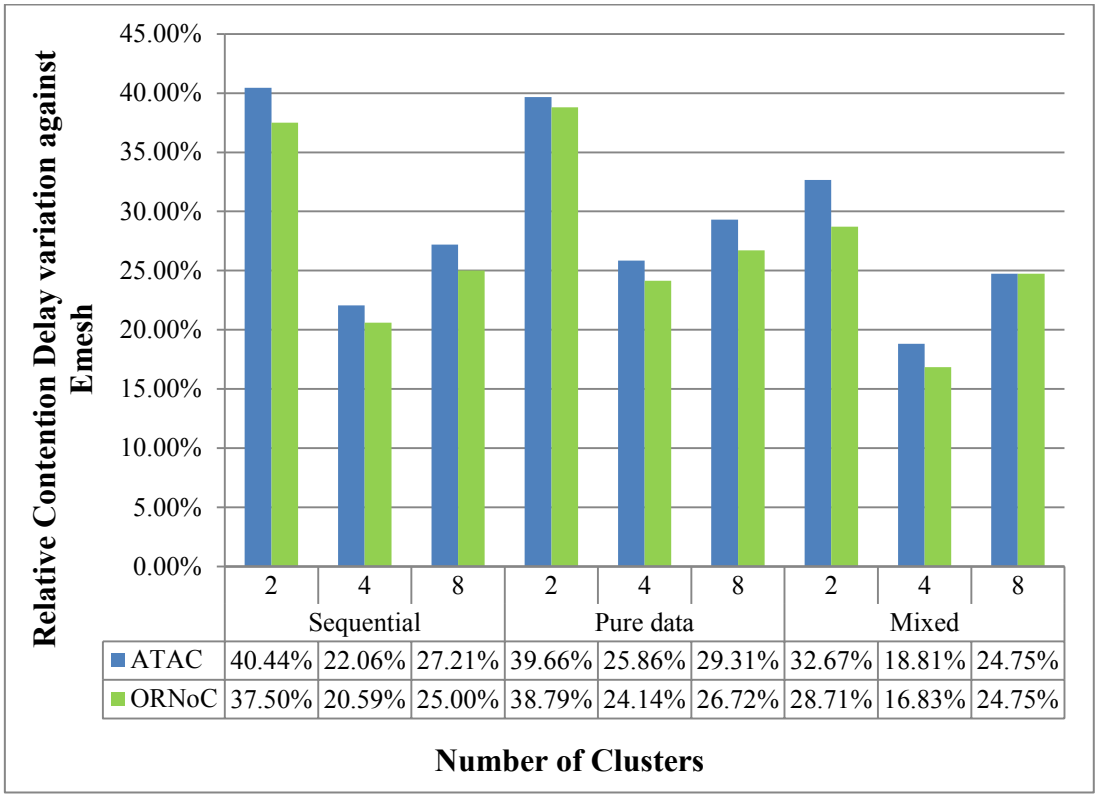
	1	2	4	8	1	2	4	8	1	2	4	8
Emesh	0.136				0.116				0.101			
ATAC		0.081	0.106	0.099		0.07	0.086	0.082		0.068	0.082	0.076
ORNoC		0.085	0.108	0.102		0.071	0.088	0.085		0.072	0.084	0.076

Figure 47: Average Contention Delay for Cluster Numbers N = {1,2,4,8}

Contention delay follows the same trend as of packet latency for different cluster sizes except for the number of cluster equal to 2, which shows a decrease in the contention delay values for ONoC architectures. When there are only 2 clusters, there is a bigger cluster size and the communication is predominantly intra cluster and hence, in the electrical network, which gives better contention values as the workload is sufficiently less. The number of optical access points per cluster has a default value of 4, and when the number of clusters is 4, the cluster size is 16. This enables more communication in the optical network, but the ratio of number of cores in a cluster to the access points is 4:1, which increases the contention in accessing them. Meanwhile, when the number of clusters increases to 8, the cluster size becomes 8 and the ratio improves to 2:1, which again reduces the contention delay. The values for contention delay are 0.136ns for sequential, 0.116ns for pure data parallelism and 0.101ns for mixed parallelism implementation in the case of Emesh architecture. The ATAC architecture shows contention delay values ranging from 0.086ns for 2 clusters to 0.099ns for 8 clusters in the case of sequential implementation, 0.07ns for 2 clusters to 0.082ns for 8 clusters for pure data parallelism implementation, 0.068ns for 2 clusters to 0.076ns for 8 clusters in the case of mixed parallelism implementation. The contention delay values for the ORNoC architecture are from 0.085ns for 2 clusters to 0.038J 0.102ns for 8 clusters in the case of sequential implementation, 0.071ns for 2 clusters to 0.085ns for 8 clusters for pure data parallelism implementation, 0.072ns for 2 clusters to 0.076ns for 8 clusters in the case of mixed parallelism implementation.

The average values for the contention delay is calculated for all the cluster sizes considered. The average values of contention delay for sequential implementation are 0.136ns, 0.095ns and 0.098ns for Emesh, ATAC and ORNoC architectures respectively. Pure data parallelism

implementation gives average values of 0.116ns for Emesh, 0.079ns for ATAC and 0.081ns for ORNoC architectures. The average values for mixed implementation are 0.101ns, 0.075ns and 0.077ns respectively for Emesh, ATAC and ORNoC architectures. The average variation between the pure data parallelism and mixed parallelism against sequential implementation are 14.71% and 25.73% for Emesh, 16.84% and 21.05% for ATAC and 17.31% and 18.37% for ORNoC respectively. Both the parallelized implementations provide improved contention delay rates than the sequential implementation due to the effective handling of data through the usage of threads. Among the parallelized implementations, the mixed parallelism implementation provides better values as it employs parallelism more effectively as compared to pure data parallelism.



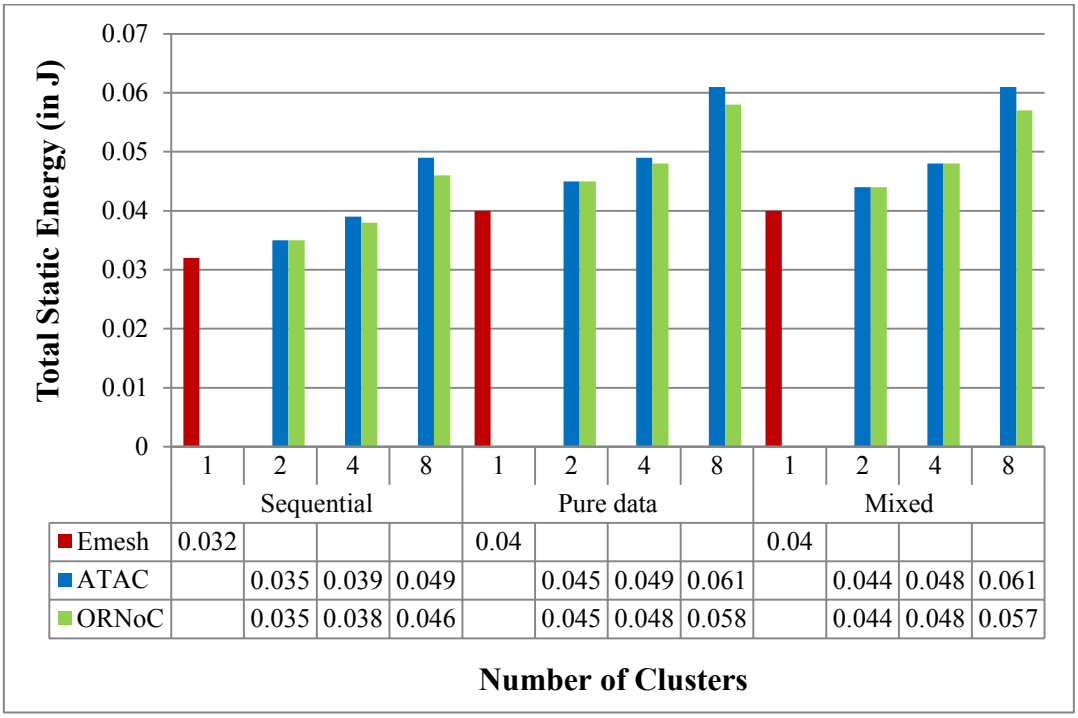
	2	4	8	2	4	8	2	4	8
	Sequential			Pure data			Mixed		
ATAC	40.44%	22.06%	27.21%	39.66%	25.86%	29.31%	32.67%	18.81%	24.75%
ORNoC	37.50%	20.59%	25.00%	38.79%	24.14%	26.72%	28.71%	16.83%	24.75%

Figure 48: Relative Contention Delay variation for Cluster Numbers N = {1,2,4,8}

The relative contention delay improvement goes from 40.44% for 2 clusters to 27.21% for 8 clusters in the case of sequential implementation, 39.66% for 2 clusters to 29.31% for 8 clusters in the case of pure data parallelism implementation and 32.67% for 2 clusters and 24.75% for 8 clusters in the case of mixed parallelism implementation for the ATAC architecture. ORNoC architecture shows relative contention delay values of 37.5% for 2 clusters to 25% for 8 clusters in the case of sequential implementation, 38.79% for 2 clusters to 26.72% for 8 clusters in the case of pure data parallelism implementation and 28.71% for 2 clusters to 15.84% for 8 clusters in the case of mixed parallelism implementation.

The average values for the relative variation is calculated for all the cluster sizes considered. The average rate of improvement in contention delay is observed to be 29.90% for ATAC and 27.7% for ORNoC for the sequential implementation, 31.61% and 29.88% for ATAC and ORNoC respectively for pure data implementation, 25.41% for ATAC and 23.43% for ORNoC in the case of mixed implementation. The Emesh architecture has higher values of contention delay than those of both the ONoC architectures due to the absence of optical network to handle communication.

5.3.3 Results for Static Energy for varying number of Clusters



	1	2	4	8	1	2	4	8	1	2	4	8
	Sequential				Pure data				Mixed			
Emesh	0.032				0.04				0.04			
ATAC		0.035	0.039	0.049		0.045	0.049	0.061		0.044	0.048	0.061
ORNoC		0.035	0.038	0.046		0.045	0.048	0.058		0.044	0.048	0.057

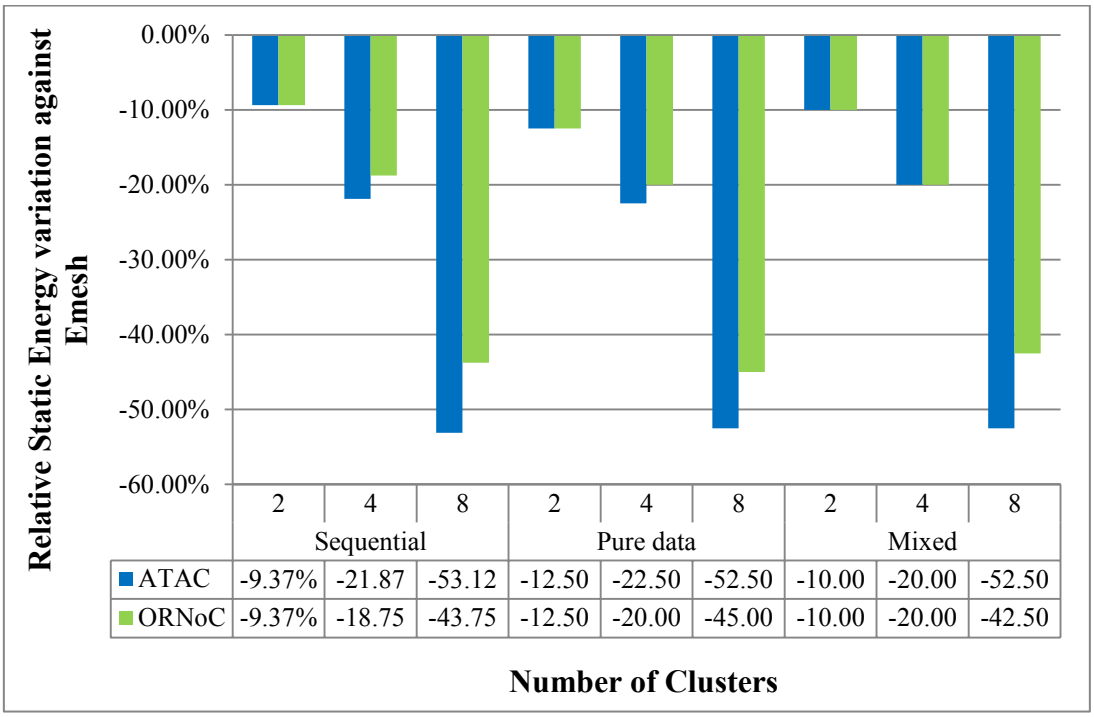
Figure 49: Total Static Energy for Cluster Numbers N = {1,2,4,8}

The total static energy consumption shows an increasing trend as the number of clusters increases. As the number of clusters increases, more optical components like hubs, receivers and detectors active, which increases the static energy consumption as their presence is the major contributor in static energy calculation. The values for static energy are 0.032J for sequential, 0.04J for both pure data parallelism and mixed parallelism implementation in the case of Emesh architecture. The ATAC architecture shows static energy values from 0.035J for 2 clusters to 0.039J for 4 clusters and 0.049J for 8 clusters in the case of sequential implementation, 0.045J for 2 clusters, 0.049J for 4 clusters and 0.061J for 8 clusters for pure data parallelism implementation, 0.044J for 2 clusters, 0.048J for 4 clusters and 0.061J for 8 clusters in the case of mixed parallelism implementation. The static energy values for the ORNoC architecture are from 0.035J for 2 clusters to 0.038J for 4 clusters and 0.046J for 8

clusters in the case of sequential implementation, 0.045J for 2 clusters, 0.048J for 4 clusters and 0.058J for 8 clusters for pure data parallelism implementation, 0.044J for 2 clusters, 0.048J for 4 clusters and 0.057J for 8 clusters in the case of mixed parallelism implementation.

The average values for the static energy is calculated for all the cluster sizes considered. The average values of static energy consumption for sequential implementation are 0.032J, 0.041J and 0.04J for Emesh, ATAC and ORNoC architectures respectively. Pure data parallelism implementation shows average values of 0.04J for Emesh, 0.052J for ATAC and 0.05J for ORNoC architectures. The average values for mixed implementation are 0.04J, 0.051J and 0.05J respectively for Emesh, ATAC and ORNoC architectures. The average variation in static energy values between the pure data parallelism and mixed parallelism against sequential implementation are both -25% for Emesh, -26.83% and -24.39% for ATAC and both -25% for ORNoC respectively, where the negative variation indicates the degeneration or increased energy consumption.

The sequential implementation gives the best results for static energy consumption while the values for both parallelized schemes are very close to each other and higher than that of sequential implementation. The number of packets in the network for sequential implementation is observed to be lesser and hence, has least static energy consumption, while it is very close to each other for both the parallelization implementations.



	2	4	8	2	4	8	2	4	8
	Sequential			Pure data			Mixed		
ATAC	-9.37%	-21.87	-53.12	-12.50	-22.50	-52.50	-10.00	-20.00	-52.50
ORNoC	-9.37%	-18.75	-43.75	-12.50	-20.00	-45.00	-10.00	-20.00	-42.50

Figure 50: Relative Static Energy variation for Cluster Numbers N = {1,2,4,8}

Both the ONoC architectures show a decline in the values with respect to Emesh with ORNoC having the lower rate of degradation. The relative degeneration rate for ATAC architecture against Emesh is -9.37% for 2 clusters and increases to -53.12% for 8 clusters in the case of sequential implementation, -12.50% for 2 clusters and increases to -52.50% for 8 clusters in the case of pure data parallelism implementation, -10% for 2 clusters and increases to -52.50% in the case of mixed parallelism implementation. ORNoC architecture shows relative degeneration rates of -9.37% for 2 clusters and increases to -43.75% for 8 clusters in the case of sequential implementation, -12.50% for 2 clusters and increases to -45% for 8 clusters in the case of pure data parallelism implementation, -10% for 2 clusters and increases to -42.50% for 8 clusters in the case of mixed parallelism implementation.

The average values for the relative variation is calculated for all the cluster sizes considered The average degradation rate is -28.12% for ATAC and -23.96% for ORNoC for sequential

implementation. The pure data parallelism implementation shows average degradation rates of -29.17% for ATAC and -25.83% for ORNoC architectures. The decline rate against Emesh is observed to be -27.5% and -24.17% for ATAC and ORNoC architectures respectively for the mixed parallelism implementation. Emesh architecture has the least static energy consumption values compared to both the ONoC architectures as the ONoC architectures have optical components like lasers, waveguides and modulators that contribute to the static energy. ORNoC shows lesser static energy consumption than ATAC in some cases as the number of wavelengths and waveguides used are lesser, which in turn reduces the number of laser sources and the losses due to thermal tuning and ring tuning [54].

5.3.4 Results for Dynamic Energy for varying number of Clusters

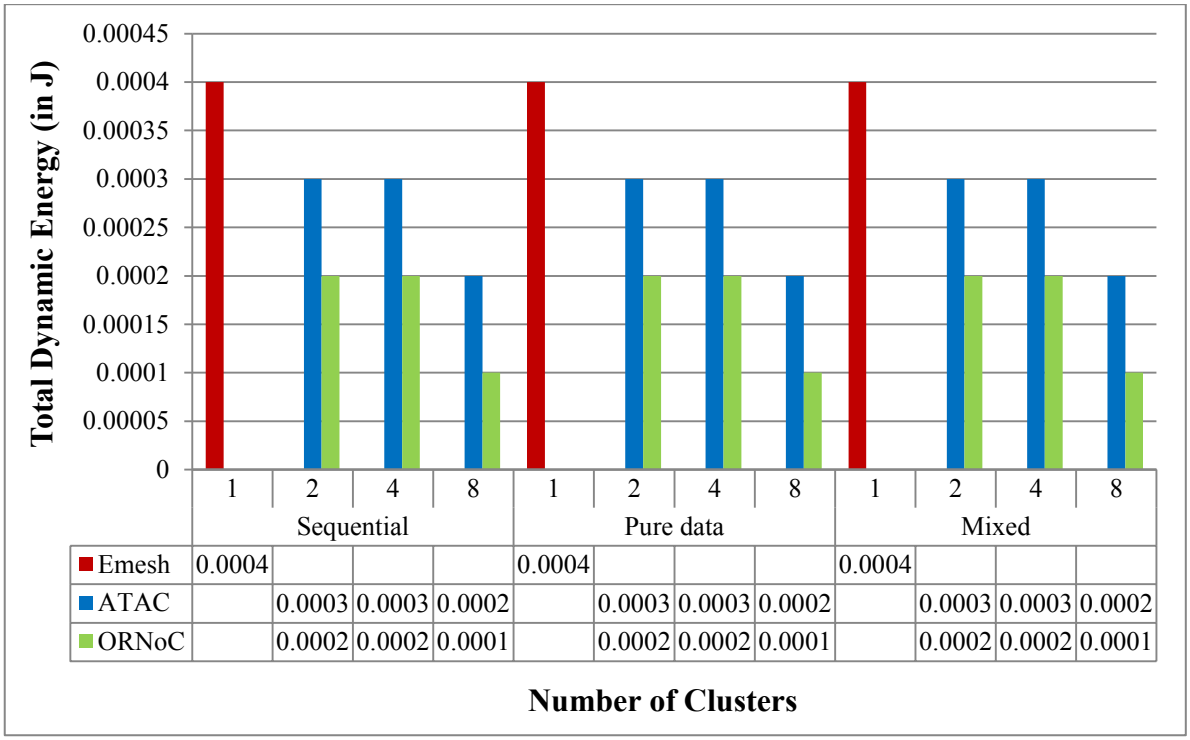


Figure 51: Total Dynamic Energy for Cluster Numbers N = {1,2,4,8}

The dynamic energy consumption for both the ONoC architectures is lesser than that of Emesh. The values decrease for the ONoC architectures with an increase in the number of clusters. The dynamic energy consumption values are observed to be 0.0004J for all implementations in the case of Emesh architecture. All the implementations show the same trend in dynamic energy consumption values for each of the network architectures. The average values for the dynamic energy is calculated for all the cluster sizes considered. The actual and average dynamic energy values are observed to be 0.0004J, 0.0003J and 0.0002J for Emesh, ATAC and ORNoC architectures for all the three implementation schemes.

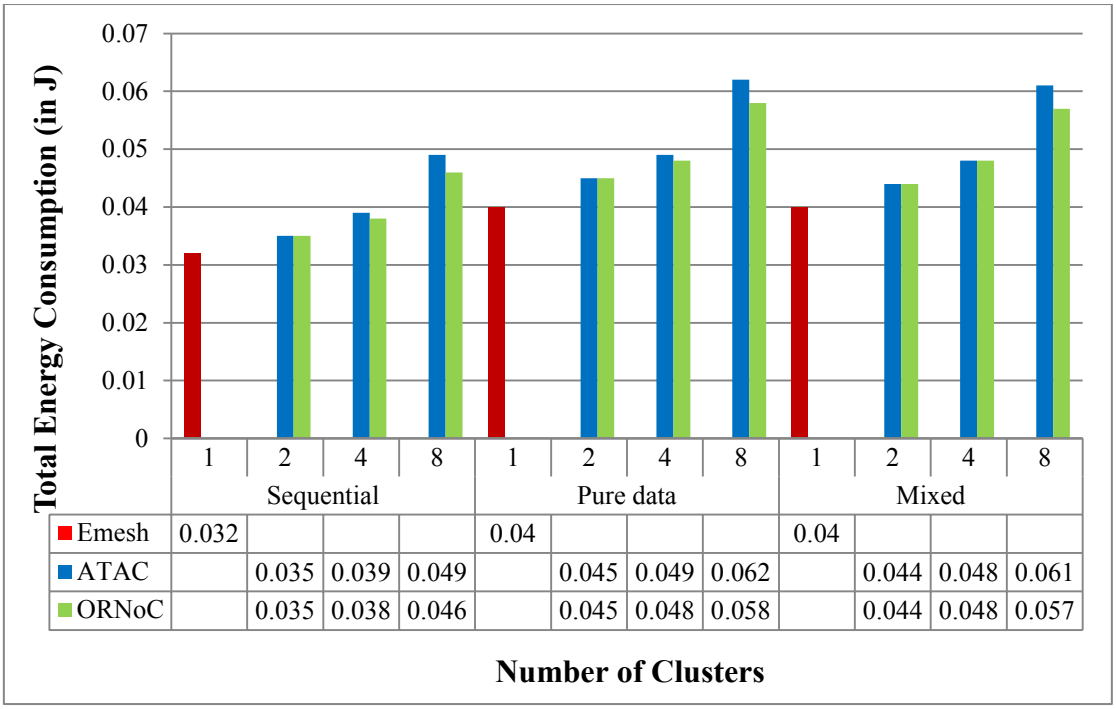
Figure 52: Relative Dynamic Energy for Cluster Numbers N = {1,2,4,8}

The relative variation in dynamic energy shows great improvement for ONoC architectures over Emesh as per Figure 52. The relative variation for dynamic energy consumption for ATAC architecture are 25% for 2 and 4 clusters while it is 50% for 8 clusters in the case of all the implementation schemes. The ORNoC architecture shows relative variation values of

50% for both 2 and 4 clusters and 75% for 8 clusters in the case of all the implementation schemes.

The average values for the relative variation is calculated for all the cluster sizes considered. The average values for dynamic energy is 33.33% for ATAC and 58.33% for ORNoC for all the implementations. ORNoC architecture shows greater improvement as it uses lesser number of receivers than ATAC. Emesh architecture is observed to have slightly higher dynamic energy consumption as compared to the ONoC architectures, because of larger number of electrical links [54]. Among the ONoC architectures, ATAC has higher energy consumption as ORNoC has the wavelength reuse property which causes a decrease in the number of receivers or modulators which constitute data dependent energy [54].

5.3.5 Results for Total Energy for varying number of Clusters



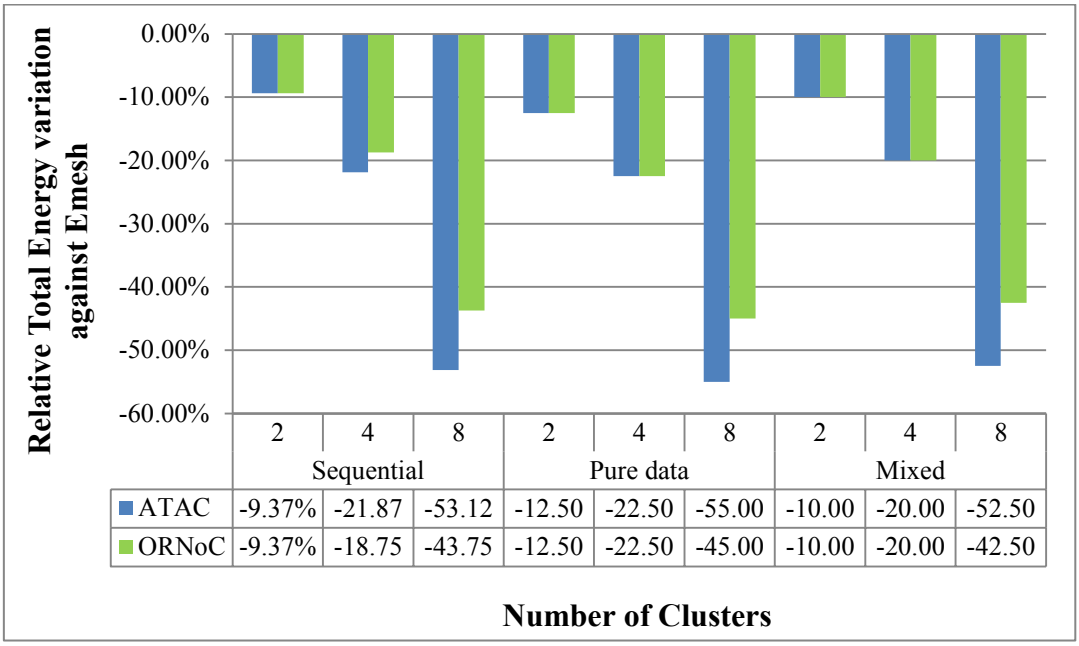
	1	2	4	8	1	2	4	8	1	2	4	8
Emesh	0.032				0.04				0.04			
ATAC		0.035	0.039	0.049		0.045	0.049	0.062		0.044	0.048	0.061
ORNoC		0.035	0.038	0.046		0.045	0.048	0.058		0.044	0.048	0.057

Figure 53: Total Energy Consumption for Cluster Numbers N = {1,2,4,8}

The total energy consumption combines both the static energy consumption as well as the dynamic energy consumption values to give the overall energy consumption in the network. The values are lowest for Emesh and the ONoC values are much closer to that of Emesh when there is lesser number of clusters. The total energy consumption values for Emesh architecture are 0.032J for the sequential implementation, 0.04J for both the pure data parallelism implementation and the mixed parallelism implementation. The total energy consumption values for the ATAC architecture increases from 0.035J for 2 clusters to 0.049J for 8 clusters in the case of sequential implementation, 0.045J for 2 clusters to 0.062J for 8 clusters in the case of pure data parallelism implementation and 0.044J for 2 clusters to 0.061J for 8 clusters in the case of mixed parallelism implementation. The ORNoC architecture shows increase in the total energy values from 0.035J for 2 clusters to 0.046J for 8 clusters for the sequential implementation, 0.045J for 2 clusters and 0.058J for 8 clusters in the case of pure data parallelism implementation and 0.044J for 2 clusters to 0.057J for 8 clusters for the mixed parallelism implementation. The energy consumption increases with increasing cluster numbers as there are more optical components. The parallelized versions of the algorithm have multiple threads that are assigned to more cores, and hence it increases the energy consumption values for the parallelized implementations. The total energy consumption gives the sum of dynamic and static energy consumption for different cores.

The average values for the total energy is calculated for all the cluster sizes considered. The average values are 0.032J, 0.041J and 0.04J for the Emesh, ATAC and ORNoC architectures respectively for the sequential implementation. Emesh architecture has 0.04J, ATAC architecture has 0.052J and ORNoC architecture has 0.05J as the average values for pure data parallelism implementation. Mixed parallelism implementation gives average values of

0.04J, 0.051J and 0.05J respectively for Emesh, ATAC and ORNoC architectures for total energy consumption. The total energy consumption increases with number of clusters due to the increase in the number of optical components, while it increases for the mixed and pure data parallelism implementations due to the increased number of threads assigned for the task.



	2	4	8	2	4	8	2	4	8
	Sequential			Pure data			Mixed		
ATAC	-9.37%	-21.87	-53.12	-12.50	-22.50	-55.00	-10.00	-20.00	-52.50
ORNoC	-9.37%	-18.75	-43.75	-12.50	-22.50	-45.00	-10.00	-20.00	-42.50

Figure 54: Relative Total Energy for Cluster Numbers N = {1,2,4,8}

The relative variation in total energy shows the quantitative results for the ONoC architectures with respect to Emesh. The relative degeneration rate for ATAC architecture against Emesh is -9.37% for 2 clusters and increases to -53.12% for 8 clusters in the case of sequential implementation, -12.50% for 2 clusters and increases to -52.50% for 8 clusters in the case of pure data parallelism implementation, -10% for 2 clusters and increases to -52.50% for 8 clusters in the case of mixed parallelism implementation. ORNoC architecture shows relative degeneration rates of -9.37% for 2 clusters and increases to -43.75% for 8 clusters in the case of sequential implementation, -12.50% for 2 clusters and increases to -

45% for 8 clusters in the case of pure data parallelism implementation, -10% for 2 clusters and increases to -42.50% for 8 clusters in the case of mixed parallelism implementation.

The average values for the relative variation is calculated for all the cluster sizes considered. The average relative variation against Emesh is -28.12% for ATAC and -23.96% for ORNoC for sequential implementation. The average degradation is observed to be -30% and -26.67% for ATAC and ORNoC respectively for pure data parallelism implementation. The decline rate for ATAC and ORNoC is -27.5% and -24.17% for mixed parallelism implementation.

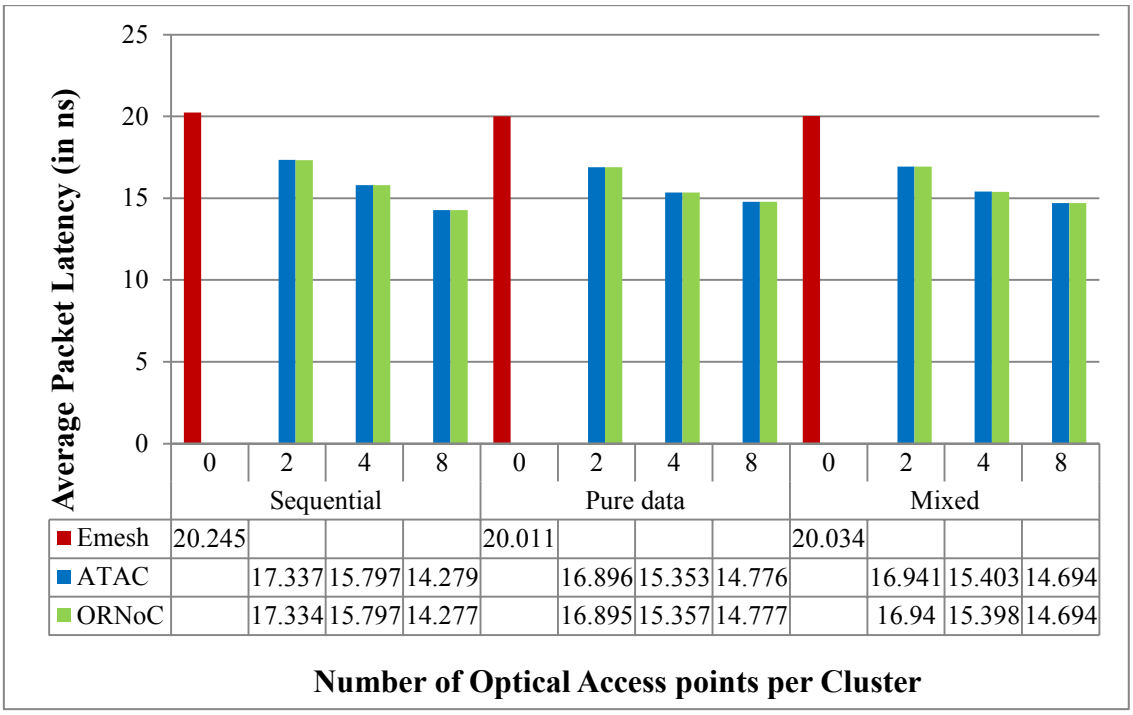
5.4 Analysis with varying Optical Access points

The number of optical access points per cluster essentially defines the number of entry and exit points to the cluster for receiving and sending the data to the optical layer. The increased number of optical access points in a cluster would reduce the use electrical layer as there would be multiple points to access the optical layer even for intra cluster communication. The increase in number of optical access points would mean that the intra cluster electrical network can be used more for actual data processing and not for routing data to the minimal number of optical hubs. The Emesh architecture has always zero optical access points per cluster as an optical network is absent. The comparison is between the different values of optical access points in ATAC and ORNoC with the Emesh architecture with zero optical access points but having full connectivity to determine the network delay and energy improvements.

The number of optical access points per cluster in the experiment considers values in the range of the cluster size of 16. This is because the number of optical access points per cluster cannot be more than the cluster size. Hence, we use a moderate cluster size, with permissible

values of optical access points. This experiment uses a square image of size 1024pixel square image executed on 64 cores. The image size is chosen as the default which is the moderate size among the sets. All the other configurations are default as mentioned in Table 4. The specific configurations are given in Appendix Table 4.

5.4.1 Results for Packet Latency for varying number of Optical Access points



	Sequential				Pure data				Mixed			
	0	2	4	8	0	2	4	8	0	2	4	8
Emesh	20.245				20.011				20.034			
ATAC		17.337	15.797	14.279		16.896	15.353	14.776		16.941	15.403	14.694
ORNoC		17.334	15.797	14.277		16.895	15.357	14.777		16.94	15.398	14.694

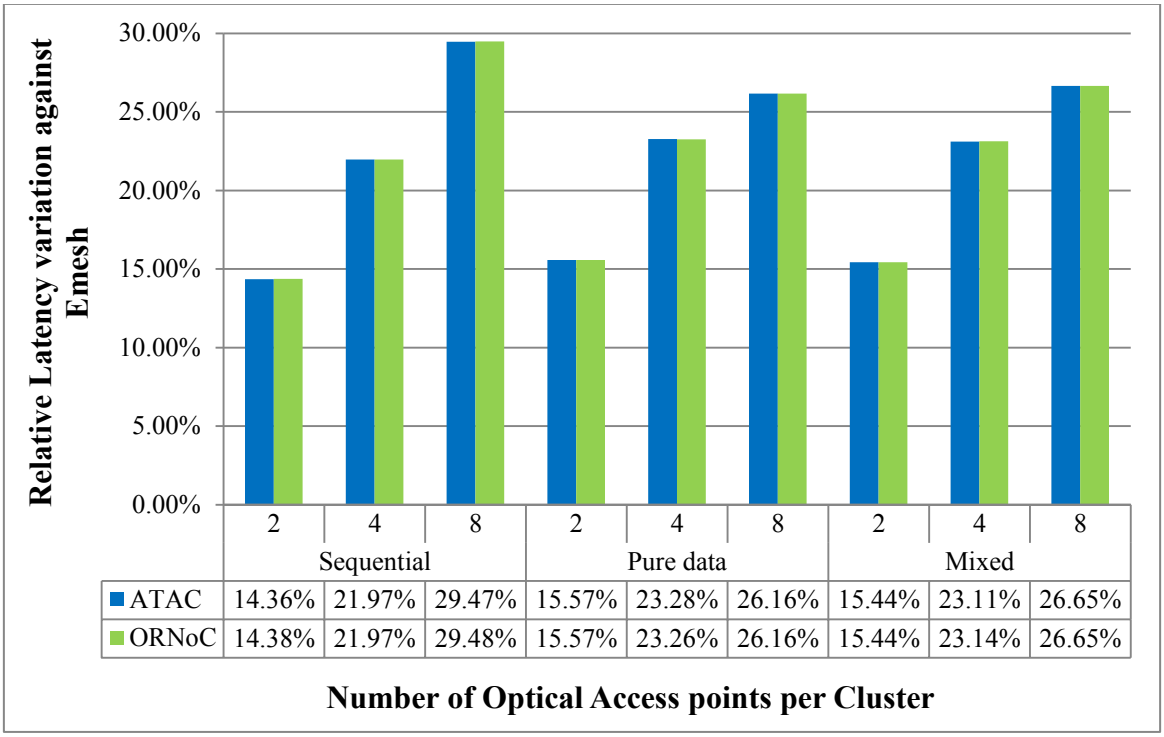
Figure 55: Average Packet Latency for Optical Access points per Cluster N = {0,2,4,8}

This figure is different form the others as Emesh has only one set of values for all the implementations while the ONoCs have all 3 sets of values. This remains the same for all the figures in this set of experiments. As the number of optical access points per cluster increases, the average packet latency decreases for all the implementations in ATAC and ORNoC which is expected as the increase in number of optical access points enables much

better intra cluster communication. The Emesh values show improvement as they go from sequential to the parallelized implementations, but as displayed in Figure 55, it has only one data point as the Emesh architecture has no optical access points, as it is a purely electrical architecture. The packet latency values for the ATAC architecture decreases from 17.337ns for 2 optical access points to 14.279ns for 8 optical access points in the case of sequential implementation, 16.896ns for 2 optical access points to 14.776ns for 8 optical access points in the case of pure data parallelism implementation and 16.941ns for 2 optical access points to 14.694ns for 8 optical access points in the case of mixed parallelism implementation. The ORNoC architecture shows a decrease in the packet latency values from 17.334ns for 2 optical access points to 14.277ns for 8 optical access points for the sequential implementation, 16.895ns for 2 optical access points and 14.777ns for 8 optical access points in the case of pure data parallelism implementation and 16.94ns for 2 optical access points to 14.694ns for 8 optical access points for the mixed parallelism implementation.

The average values for the packet latency is calculated for all the optical access points considered. The packet latency values for Emesh architecture are 20.245ns for the sequential implementation, 20.011ns for the pure data parallelism implementation and 20.034ns for the mixed parallelism implementation. The average improvement in the packet latency values are 20.245ns, 15.804ns and 15.803ns for Emesh, ATAC and ORNoC architectures respectively in the case of sequential implementation. The pure data parallelism implementation has 20.011ns for Emesh, 15.675ns for ATAC and 15.676ns for ORNoC as the average latency values. The average values for packet latency are 20.034ns for Emesh, 15.679ns for ATAC and 15.677ns for ORNoC in the case of mixed implementation. The average variation in the latency values against sequential implementation are 1.16% and

1.04% for Emesh, 0.82% and 0.79% for ATAC, 0.80% for both cases in ORNoC in the case of pure data parallelism and mixed implementation schemes respectively. All the implementations provide very similar values in the case of packet latency with negligible variations indicating that the improvement in packet latency is uniform for this experiment.



	Sequential			Pure data			Mixed		
	2	4	8	2	4	8	2	4	8
ATAC	14.36%	21.97%	29.47%	15.57%	23.28%	26.16%	15.44%	23.11%	26.65%
ORNoC	14.38%	21.97%	29.48%	15.57%	23.26%	26.16%	15.44%	23.14%	26.65%

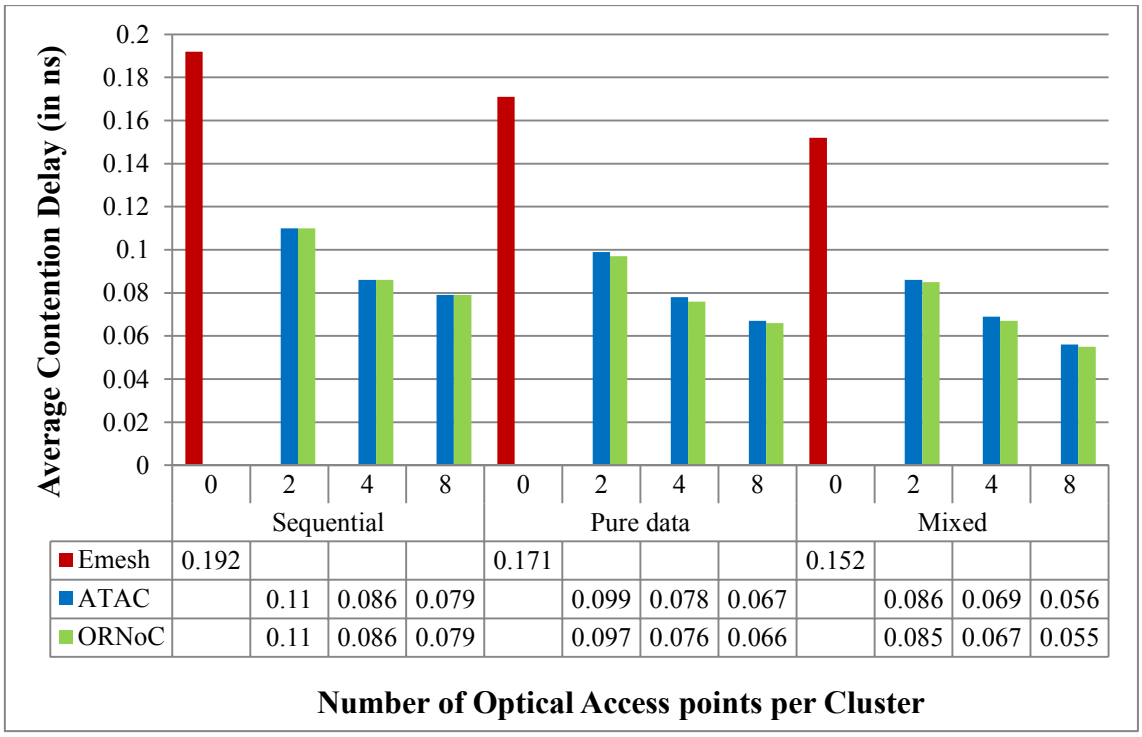
Figure 56: Relative Latency variation for Optical Access points per Cluster N = {0,2,4,8}

The relative variation with respect to Emesh graph shows that the latency values improve with the increase in the number of optical access points per cluster for all the parallelization schemes. The relative improvement rate for packet latency in the case of ATAC architecture against Emesh is 14.36% for 2 optical access points and increases to 29.47% for 8 optical access points in the case of sequential implementation, 15.57% for 2 clusters and increases to 26.16% for 8 clusters in the case of pure data parallelism implementation, 15.44% for 2 clusters and increases to 26.65% for 8 optical access points in the case of mixed parallelism implementation. ORNoC architecture shows relative degeneration rates of 14.38% for 2

optical access points and increases to 29.48% for 8 optical access points in the case of sequential implementation, 15.57% for 2 optical access points and increases to 26.16% for 8 optical access points in the case of pure data parallelism implementation, 15.44% for 2 optical access points and increases to 26.65% for 8 optical access points in the case of mixed parallelism implementation.

The average values for the relative variation is calculated for all the optical access points considered. The sequential implementation gives an average improvement rate of 21.93% for ATAC and 21.94% for ORNoC. The pure data parallelism implementation gives average improvement of 21.67% and 21.66% for ATAC and ORNoC architectures respectively. ATAC and ORNoC architectures show average relative variation values of 21.73% and 21.74% for the mixed parallelism implementation. Emesh performs much worse than the ONoC architectures due to the absence of optical connectivity.

5.4.2 Results for Contention Delay for varying number of Optical Access points



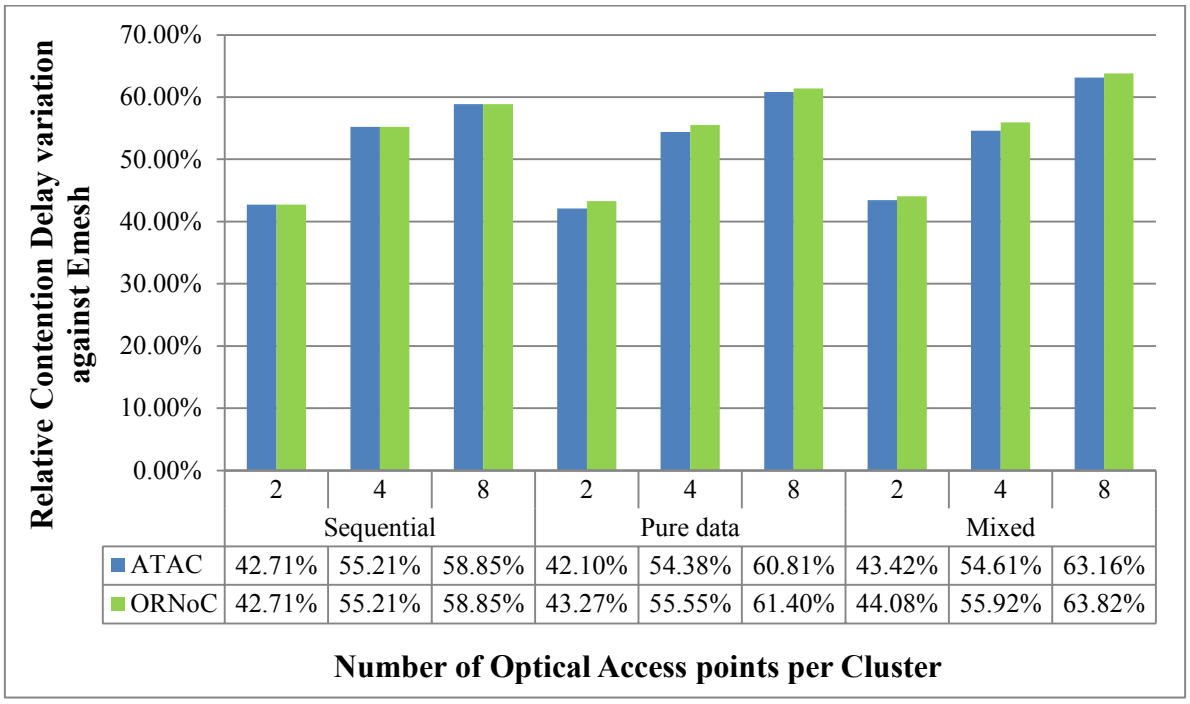
	0	2	4	8	0	2	4	8	0	2	4	8
	Sequential				Pure data				Mixed			
Emesh	0.192				0.171				0.152			
ATAC		0.11	0.086	0.079		0.099	0.078	0.067		0.086	0.069	0.056
ORNoC		0.11	0.086	0.079		0.097	0.076	0.066		0.085	0.067	0.055

Figure 57: Average Contention Delay for Optical Access points per Cluster N = {0,2,4,8}

Contention delay values show that they decrease with the number of optical access points per cluster. The increase in access points leads to more entry points to the optical network which in turn reduces the waiting time for data packets to access the optical network. The contention delay values for Emesh architecture are from 0.192ns for sequential implementation, 0.171ns for the pure data implementation and 0.152ns for the mixed parallelism implementation. The contention delay values for the ATAC architecture decreases from 0.11ns for 2 optical access points to 0.079ns for 8 optical access points in the case of sequential implementation, 0.099ns for 2 optical access points to 0.067ns for 8 optical access points in the case of pure data parallelism implementation and 0.086ns for 2 optical access points to 0.056ns for 8 optical access points in the case of mixed parallelism

implementation. The ORNoC architecture shows a decrease in the contention delay values from 0.11ns for 2 optical access points to 0.079ns for 8 optical access points for the sequential implementation, 0.097ns for 2 optical access points and 0.066ns for 8 optical access points in the case of pure data parallelism implementation and 0.085ns for 2 optical access points to 0.055ns for 8 optical access points for the mixed parallelism implementation.

The average values for the contention delay is calculated for all the optical access points considered. The average values of contention delay for sequential implementation are 0.192ns for Emesh and 0.092ns for both ATAC and ORNoC architectures respectively. Pure data parallelism implementation gives average values of 0.171ns for Emesh, 0.081ns for ATAC and 0.08ns for ORNoC architectures. The average values for mixed implementation are 0.152ns, 0.07ns and 0.069ns respectively for Emesh, ATAC and ORNoC architectures. The average variation between the pure data parallelism and mixed parallelism against sequential implementation are 10.94% and 20.83% for Emesh, 11.96% and 23.91% for ATAC and 13.04% and 25% for ORNoC respectively. The mixed parallelism implementation exhibits the best variation in contention delay compared to all the three implementations followed by the pure data parallelism implementation due to the effective communication using multiple threads even in the application level.



	2	4	8	2	4	8	2	4	8
	Sequential			Pure data			Mixed		
ATAC	42.71%	55.21%	58.85%	42.10%	54.38%	60.81%	43.42%	54.61%	63.16%
ORNoC	42.71%	55.21%	58.85%	43.27%	55.55%	61.40%	44.08%	55.92%	63.82%

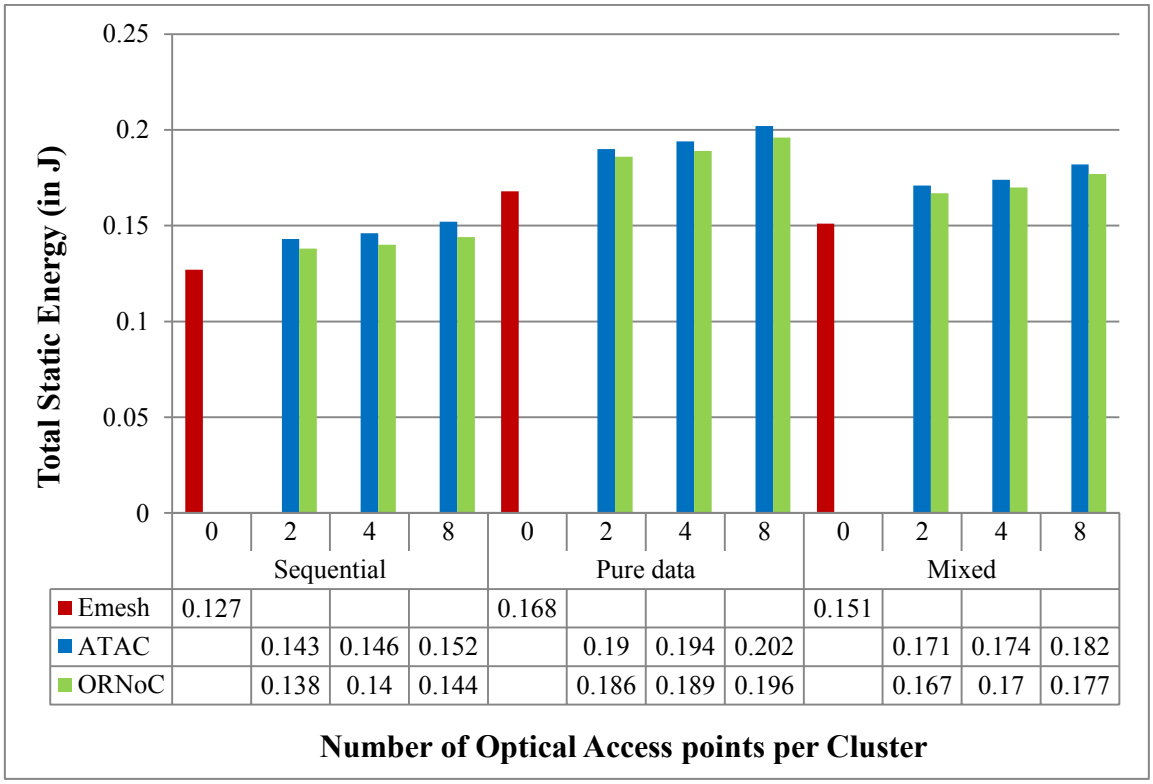
Figure 58: Relative Contention Delay variation for Optical Access points per Cluster N = {0,2,4,8}

The improvement in contention delay in relation to Emesh increases with increase in optical access points per cluster. The relative improvement rate for contention delay in the case of ATAC architecture against Emesh is 42.71% for 2 optical access points and increases to -58.85% for 8 optical access points in the case of sequential implementation, 42.10% for 2 optical access points and increases to 60.81% for 8 optical access points in the case of pure data parallelism implementation, 43.42% for 2 optical access points and increases to 63.16% in the case of mixed parallelism implementation. ORNoC architecture shows relative improvement rates of 42.71% for 2 optical access points and increases to 58.85% for 8 optical access points in the case of sequential implementation, 43.27% for 2 optical access points and increases to 61.40% for 8 optical access points in the case of pure data parallelism

implementation, 44.08% for 2 optical access points and increases to 63.82% for 8 optical access points in the case of mixed parallelism implementation.

The average values for the relative variation is calculated for all the optical access points considered. The average values for improvement is 52.25% for both ATAC and ORNoC in the case of sequential implementation. The average improvement in the case of pure data parallelism implementation is 47.76% for both ATAC and ORNoC architectures. The average improvement in the case of mixed parallelism implementation is 53.73% and 54.61% for ATAC and ORNoC respectively. The ONoC architectures perform much better than Emesh due to the presence of optical network which considerably reduces the contention delay. Also, the number of lesser number of waveguides and wavelengths for ORNoC enables it to perform slightly better than ATAC in the case of mixed parallelism implementation.

5.4.3 Results for Static Energy for varying number of Optical Access points



	0	2	4	8	0	2	4	8	0	2	4	8
Emesh	0.127				0.168				0.151			
ATAC		0.143	0.146	0.152		0.19	0.194	0.202		0.171	0.174	0.182
ORNoC		0.138	0.14	0.144		0.186	0.189	0.196		0.167	0.17	0.177

Figure 59: Static Energy for Optical Access points per Cluster N = {0,2,4,8}

Static energy consumption in the network increases with increasing number of optical access points per cluster. As the number of optical access points causes an increase in the number of optical components in the network, it leads to increased static energy consumption. . The values for static energy consumption are 0.127J for sequential, 0.168J for pure data parallelism and 0.151J for mixed parallelism implementation in the case of Emesh architecture. The ATAC architecture shows static energy values from 0.143J for 2 optical access points and 0.152J for 8 optical access points in the case of sequential implementation, 0.19J for 2 optical access points and 0.202J for 8 optical access points for pure data parallelism implementation, 0.171J for 2 optical access points and 0.182J for 8 optical access

points in the case of mixed parallelism implementation. The static energy consumption values for the ORNoC architecture are from 0.138J for 2 optical access points to 0.144J for 8 optical access points in the case of sequential implementation, 0.186J for 2 optical access points and 0.196J for 8 optical access points for pure data parallelism implementation, 0.167J for 2 optical access points and 0.177J for 8 optical access points in the case of mixed parallelism implementation.

The average values for the static energy is calculated for all the optical access points considered. The average values of static energy consumption for sequential implementation are 0.127J, 0.147J and 0.141J for Emesh, ATAC and ORNoC architectures respectively. Pure data parallelism implementation shows average values of 0.168J for Emesh, 0.195J for ATAC and 0.190J for ORNoC architectures. The average values for mixed implementation are 0.151J, 0.176J and 0.171J respectively for Emesh, ATAC and ORNoC architectures. The average variation in static energy values between the pure data parallelism and mixed parallelism against sequential implementation are -32.28% and -18.9% for Emesh, -32.65% and -19.73% for ATAC and -34.75% and -21.28% for ORNoC respectively, where the negative variation indicates the degeneration or increased energy consumption. The pure data implementation is observed to have maximum number of packets in the network, followed by the mixed implementation and then by the sequential implementation which causes an increase in the data independent components in the same order and therefore, for the static energy consumption.

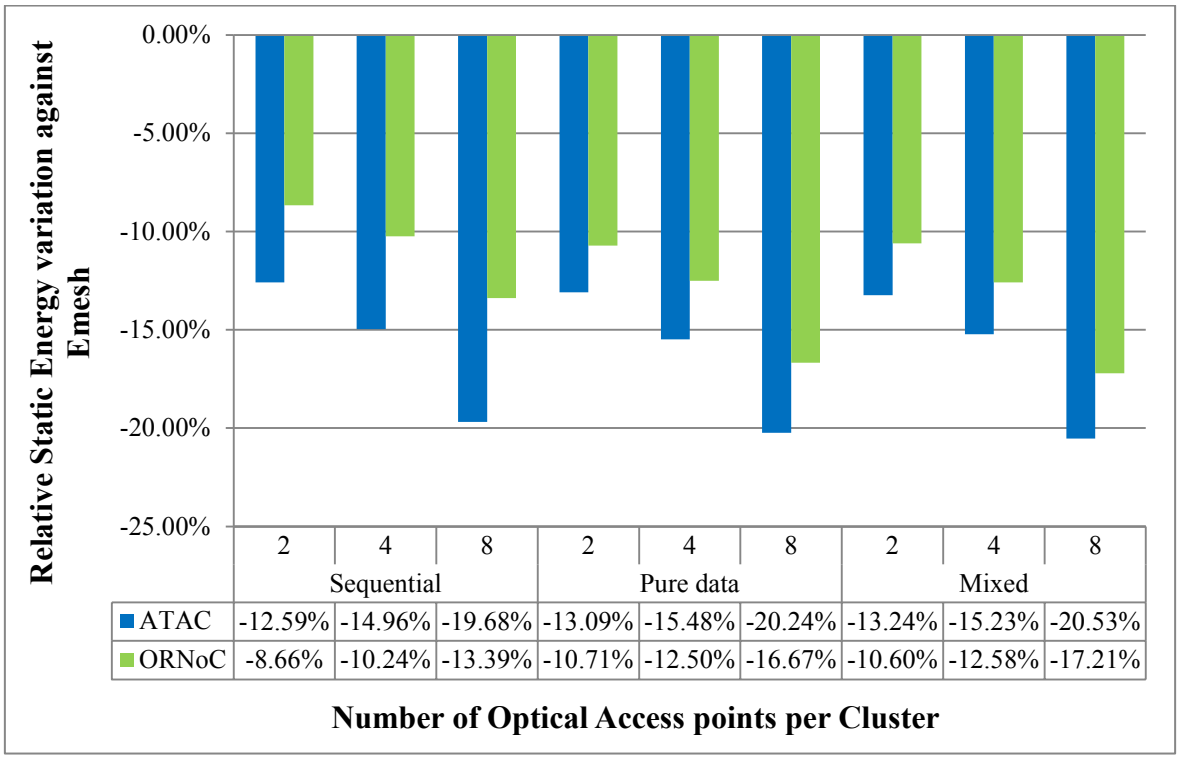


Figure 60: Relative Static Energy variation for Optical Access points per Cluster N = {0,2,4,8}

The relative static energy consumption variation against Emesh increases with increasing optical access points per cluster. The relative degeneration rate for contention delay in the case of ATAC architecture against Emesh is -12.59% for 2 optical access points and increases to -19.68% for 8 optical access points in the case of sequential implementation, -13.09% for 2 optical access points and increases to -20.24% for 8 optical access points in the case of pure data parallelism implementation, -13.24% for 2 optical access points and increases to -20.53% in the case of mixed parallelism implementation. ORNoC architecture shows relative degradation rates of -8.66% for 2 optical access points and increases to -13.39% for 8 optical access points in the case of sequential implementation, -10.71% for 2 optical access points and increases to -16.67% for 8 optical access points in the case of pure

data parallelism implementation, -10.6% for 2 optical access points and increases to -17.21% for 8 optical access points in the case of mixed parallelism implementation.

The average values for the relative variation is calculated for all the optical access points considered. The average decline rates against Emesh are -15.74% and -10.76% for ATAC and ORNoC respectively for the sequential implementation, -16.27% for ATAC and -13.29% for ORNoC for the pure data implementation, -16.33% and -13.46% for ATAC and ORNoC respectively for the mixed implementation. The decline rate for ORNoC is smaller compared to ATAC. The values of static energy consumption are minimal for Emesh architecture, since it has only electrical components [54]. Among the optical network architectures, ORNoC outperforms ATAC for all the implementations of the application because the number of wavelengths and modulators that require electrical circuitry are comparatively lesser for ORNoC due to wavelength reuse [54].

5.4.4 Results for Dynamic Energy for varying number of Optical Access points

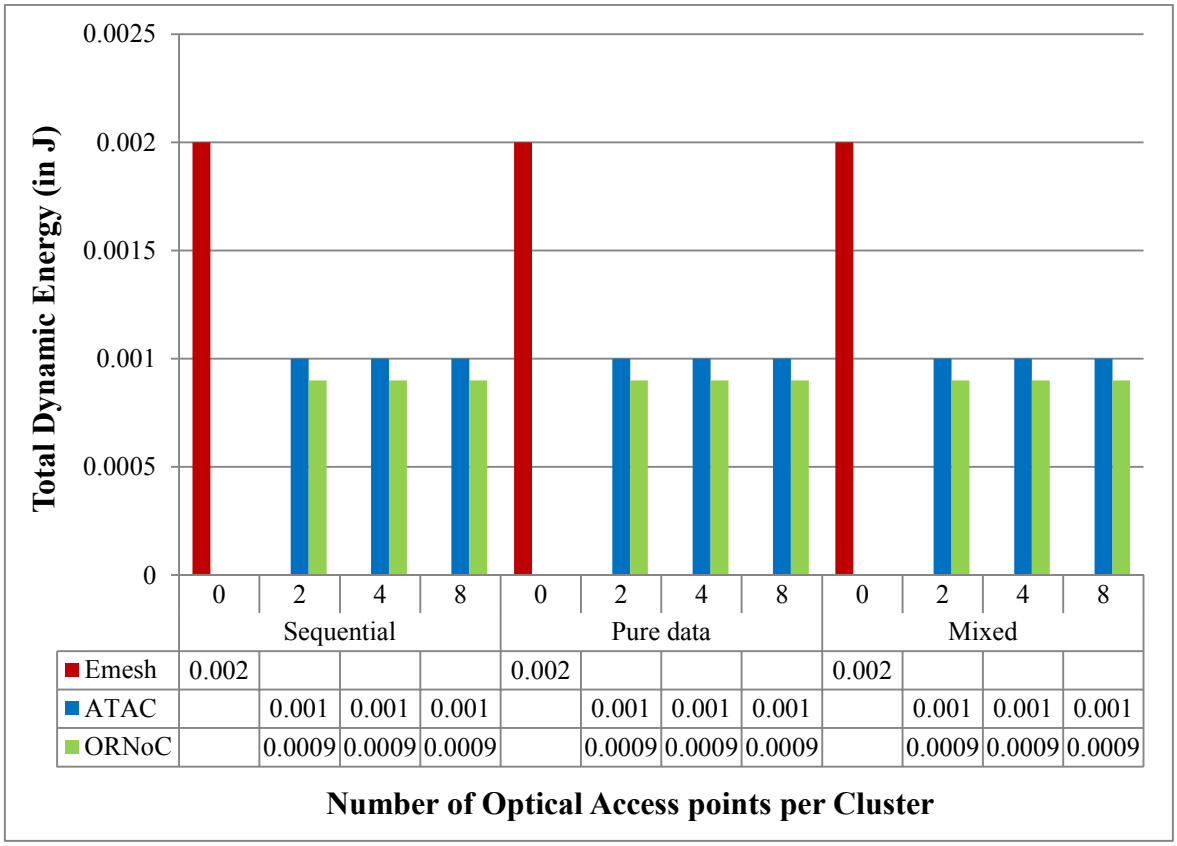


Figure 61: Dynamic Energy consumption for Optical Access points per Cluster N = {0,2,4,8}

The dynamic energy consumption values remain constant for each of the architectures with the increase in optical access points. They also remain constant for all the parallelization schemes implemented in this experiment. The actual and average values are 0.002J for Emesh and 0.001J for the ATAC and 0.0009J for the ORNoC architectures. The average values for the dynamic energy is calculated for all the optical access points considered. Emesh has slightly higher dynamic energy consumption as compared to both the ONoC architectures. The values are observed to be similar for all the implementation schemes for

each of the architectures. The dynamic energy consumption is observed to be the same for all the implementations as the workload remains the same for the experiment.

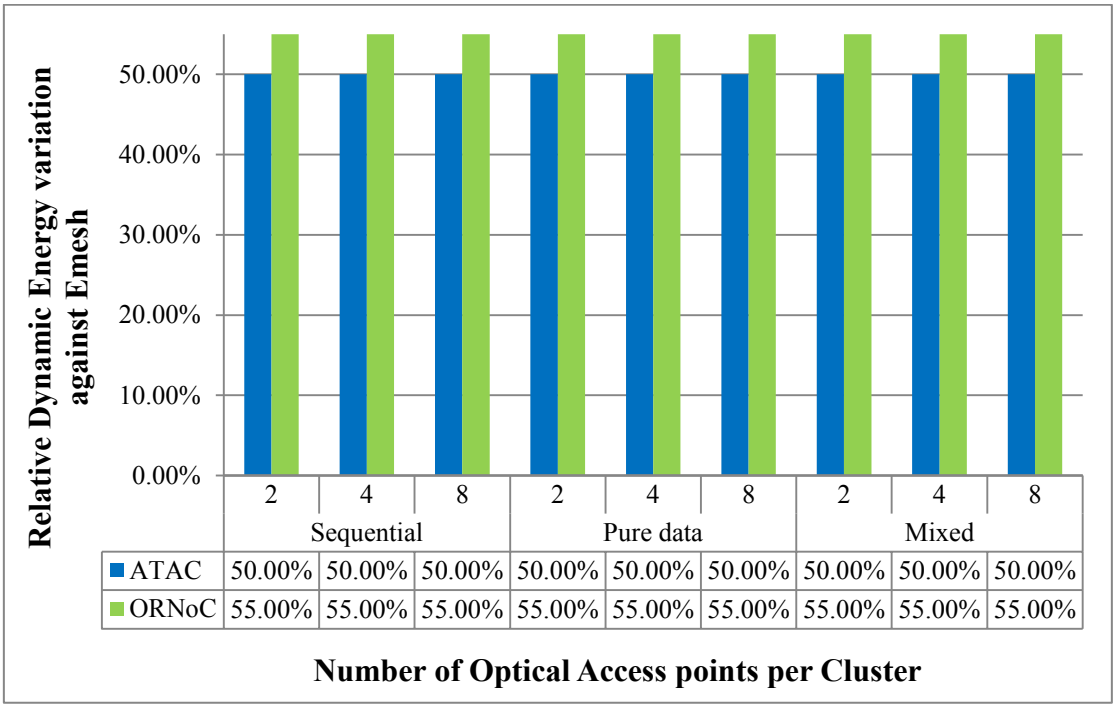
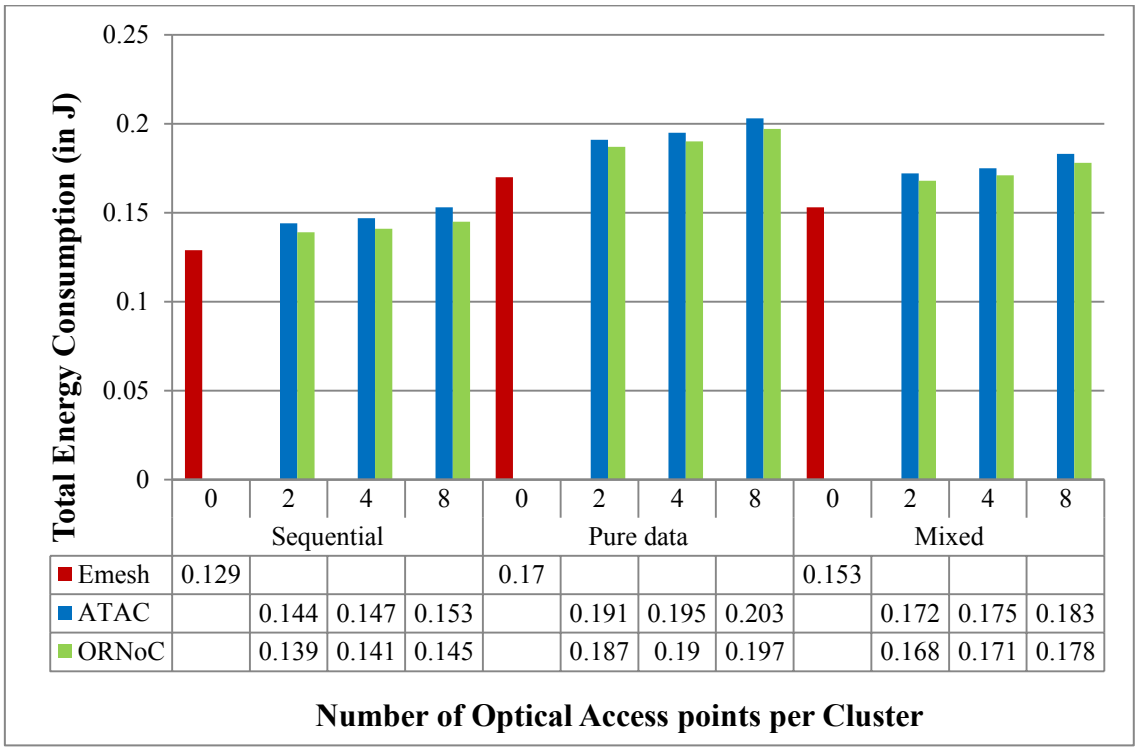


Figure 62: Relative Dynamic Energy variation for Optical Access points per Cluster N = {0,2,4,8}

The relative variation in dynamic energy consumption against Emesh remains the same for both the ONoC architectures irrespective of the optical access points per cluster considered and the implementation scheme. The actual and average relative variation values for ATAC shows an improvement of 50% while ORNoC shows an improvement of 55% in dynamic energy consumption over Emesh architecture for all the implementations. The average values for the relative variation is calculated for all the optical access points considered. Emesh architecture is observed to have slightly higher dynamic energy consumption as compared to the ONoC architectures, because of larger number of electrical links [54]. Among the ONoC architectures, ATAC has higher energy consumption as ORNoC can reuse

wavelengths, which causes a decrease in the number of receivers or modulators which constitute data dependent energy [54].

5.4.5 Results for Dynamic Energy for varying number of Optical Access points



	0	2	4	8	0	2	4	8	0	2	4	8
Emesh	0.129				0.17				0.153			
ATAC		0.144	0.147	0.153		0.191	0.195	0.203		0.172	0.175	0.183
ORNoC		0.139	0.141	0.145		0.187	0.19	0.197		0.168	0.171	0.178

Figure 63: Total Energy Consumption for Optical Access points per Cluster N = {0,2,4,8}

The total energy consumption values increase with increasing optical access points. The total energy consumption follows the same trend as that of static energy as the dynamic energy values are observed to be constant for all the cases. The values for total energy consumption are 0.129J for sequential, 0.17J for pure data parallelism and 0.153J for mixed parallelism implementation in the case of Emesh architecture. The ATAC architecture shows total energy values from 0.144J for 2 optical access points to 0.147J for 4 optical access points and 0.153J for 8 optical access points in the case of sequential implementation, 0.191J for 2

optical access points, 0.195J for 4 optical access points and 0.203J for 8 optical access points for pure data parallelism implementation, 0.172J for 2 optical access points, 0.175J for 4 optical access points and 0.183J for 8 optical access points in the case of mixed parallelism implementation. The total energy consumption values for the ORNoC architecture are from 0.139J for 2 optical access points to 0.141J for 4 optical access points and 0.145J for 8 optical access points in the case of sequential implementation, 0.187J for 2 optical access points, 0.19J for 4 optical access points and 0.197J for 8 optical access points for pure data parallelism implementation, 0.168J for 2 optical access points, 0.171J for 4 optical access points and 0.178J for 8 optical access points in the case of mixed parallelism implementation.

The average values for the total energy is calculated for all the optical access points considered. The average values for total energy consumption for the sequential implementation are 0.129J, 0.148J and 0.142J for the Emesh, ATAC and ORNoC architectures respectively. The pure data parallelism implementation shows average values of 0.17J for Emesh, 0.196J for ATAC and 0.191J for ORNoC architectures. Emesh shows average values of 0.153J, ATAC shows 0.177J and 0.172J for ORNoC for mixed parallelism implementations. The different implementations also follow the same trend as in the case of static energy consumption.

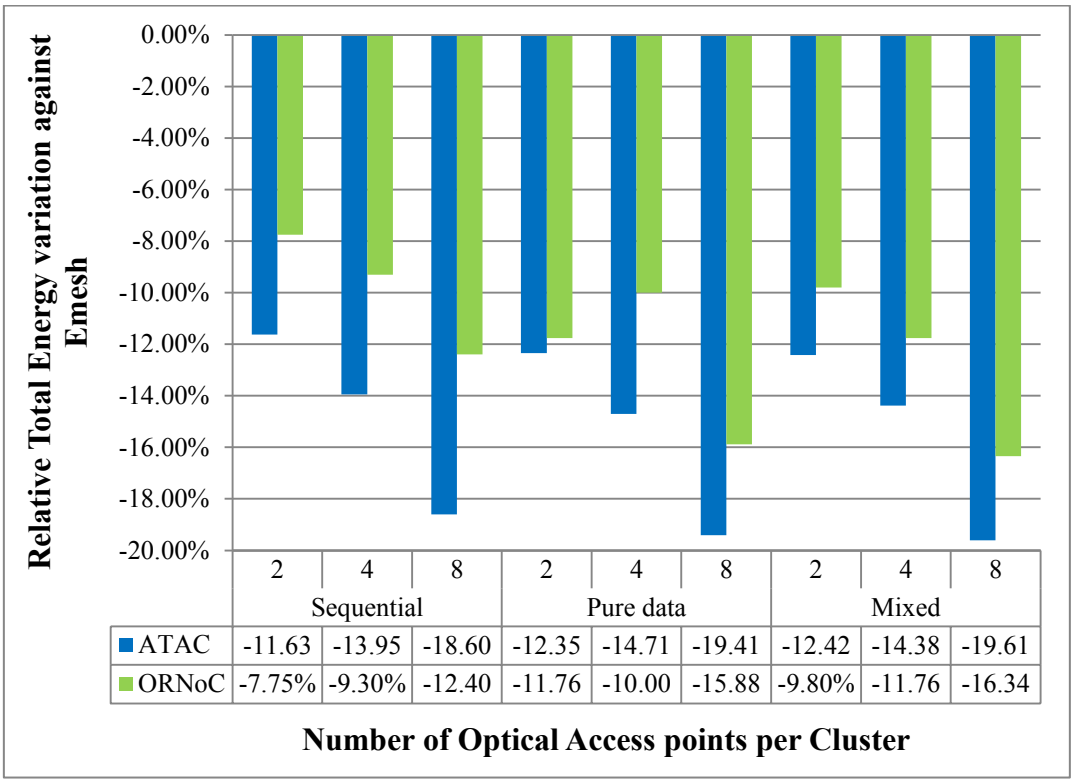


Figure 64: Relative Total Energy variation for Optical Access points per Cluster N = {0,2,4,8}

The relative variation portrays the trend in decline for the total energy consumption values against Emesh. The decline rates for ATAC architecture are observed to be -11.63% for 2 optical access points and increases to -18.63% for 8 optical access points in the case of sequential implementation, -12.35% for 2 optical access points and increases to -19.41% for 8 clusters in the case of pure data parallelism implementation, -12.42% for 2 optical access points and increases to -19.61% for 8 clusters in the case of mixed parallelism implementation. For ORNoC architecture, the decline rates are observed to be -7.75% for 2 optical access points and increases to –12.40% for 8 optical access points in the case of sequential implementation, -11.76% for 2 optical access points and increases to -15.88% for 8 clusters in the case of pure data parallelism implementation, -9.8% for 2 optical access

points and increases to -16.34% for 8 clusters in the case of mixed parallelism implementation.

The average values for the relative variation is calculated for all the optical access points considered. The average decline rates are observed to be -14.73% for ATAC and -9.82% for ORNoC for sequential, -15.49% and -12.55% for ATAC and ORNoC respectively for pure data parallelism and -15.47% for ATAC and -12.63% for ORNoC for the mixed parallelism implementations.

5.5 Analysis with varying Routing strategies

The ONoC architectures considered for our simulations support cluster based and distance based routing strategies to move data packets through the network. The cluster based routing scheme always uses the optical network for inter cluster communication while the distance based scheme considers a threshold distance, below which the packet is transmitted over electrical network and above which optical communication is used.

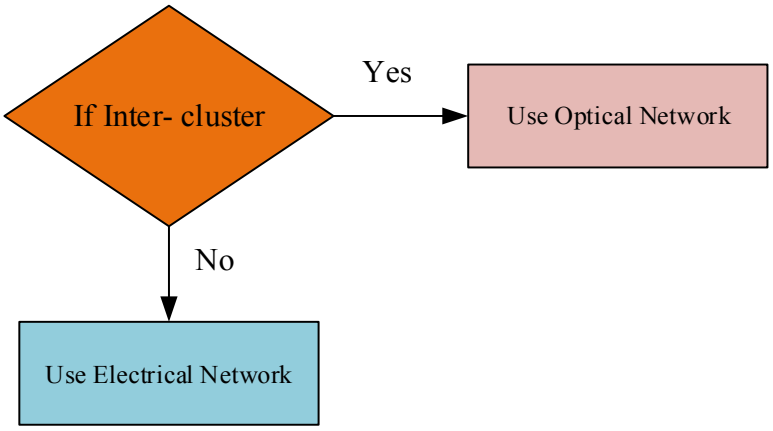


Figure 65: Cluster based Routing Topology

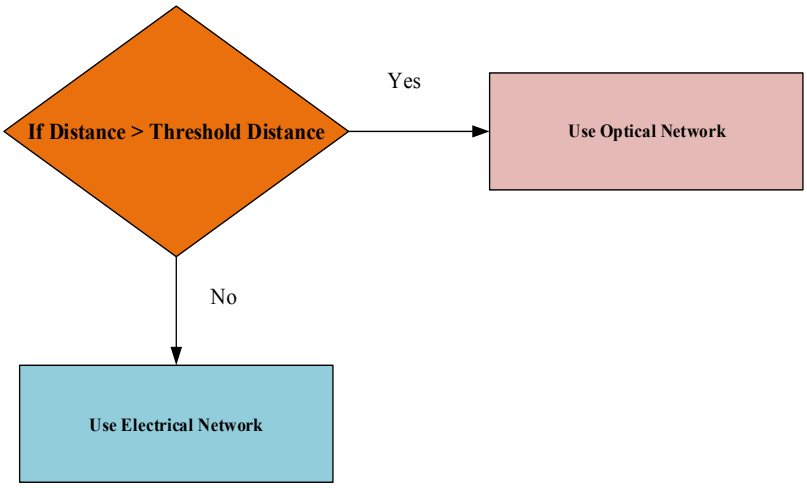


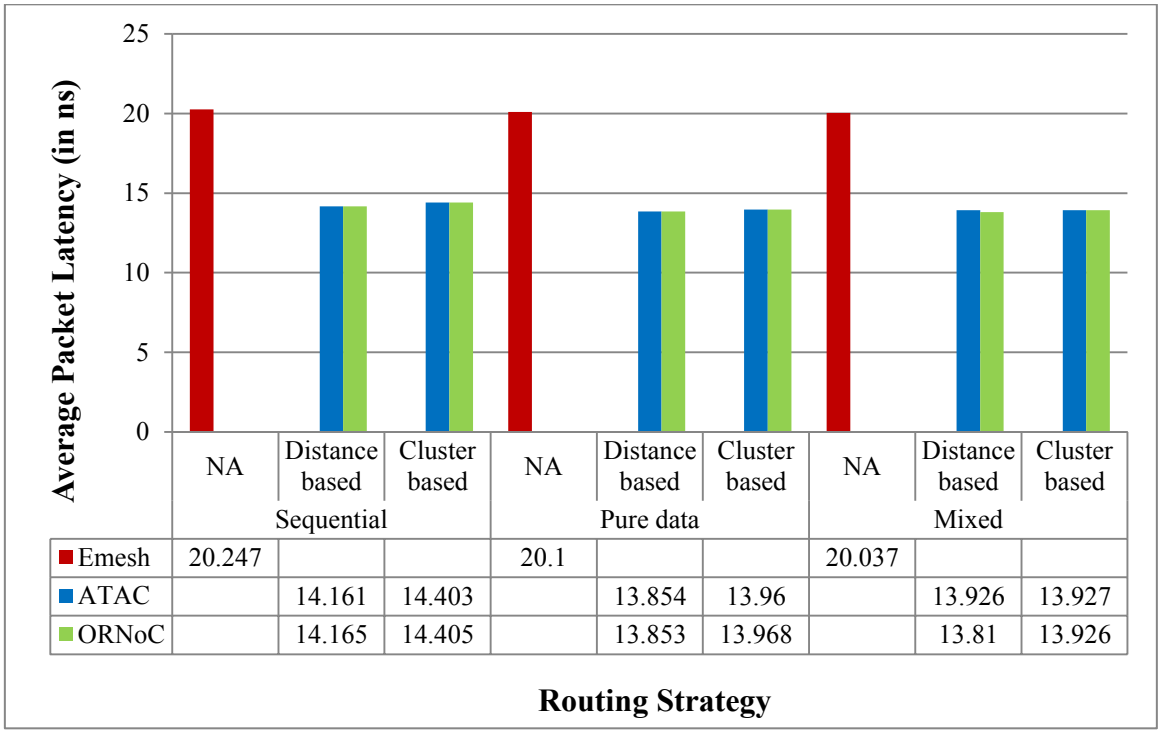
Figure 66: Distance based Routing Topology

The default value for distance threshold is 4 links or hops according to the configuration file. The Emesh network has only electrical interconnections and hence there is no specific choice of routing strategy applicable but it can be quite accurately approximated that it has distance based communication with the threshold being the distance between the first and last node. Also, according to the approximation from section 5.3, Emesh architecture has a cluster size which is equal to the number of cores. So, in order for comparison, this experiment will also consider the above approximation.

The effectiveness of cluster based and distance based routing is compared for the same load with the default distance threshold. The experiments which analyze the results with changing global routing strategy uses 64 cores and a cluster size of 8. Cluster size 8 allows for fair comparison between the two routing strategies for the threshold distance used and ensures fair amount of communication in the optical network for the distance based routing strategy. An average sized workload is chosen in the form of the image of size 1024 x 1024 pixels for the chosen number of cores. All the plots which portray the actual values of the outputs have only one value for Emesh architecture as it has only one specific routing strategy which

cannot be altered. All the other configurations are default as mentioned in Table 4. The specific configurations are given in the 5.The Emesh architecture uses only one XY routing while the optical architectures have cluster based and distance-based routing strategies. So, the graphs for the experiments in this section are similar to the ones in the previous section.

5.5.1 Results for Packet Latency for varying Routing strategies



	Sequential			Pure data			Mixed		
	NA	Distance based	Cluster based	NA	Distance based	Cluster based	NA	Distance based	Cluster based
Emesh	20.247			20.1			20.037		
ATAC		14.161	14.403		13.854	13.96		13.926	13.927
ORNoC		14.165	14.405		13.853	13.968		13.81	13.926

Figure 67: Average Packet Latency for Routing Strategies N = {distance, cluster}

The average packet latency values are observed to be slightly greater for cluster based routing for each of the implementations. Emesh architecture shows packet latency variations of 20.247ns for the sequential implementation, 20.1ns for the pure data parallelism implementation and 20.037ns for the mixed parallelism implementation. The packet latency values for the ATAC architecture are 14.161ns for the distance based routing and 14.403ns for the cluster based routing strategy in the case of sequential implementation, 13.854ns for the distance based routing and 13.96ns for the cluster based routing strategy in the case of

pure data parallelism and 13.926ns for the distance based routing and 13.927ns for the cluster based routing strategy in the case of mixed parallelism implementation. The ORNoC architecture shows packet latency variation of 14.165ns for the distance based routing and 14.405ns for the cluster based routing strategy in the case of sequential implementation, 13.853ns for the distance based routing and 13.968ns for the cluster based routing strategy in the case of pure data parallelism and 13.81ns for the distance based routing and 13.926ns for the cluster based routing strategy in the case of mixed parallelism implementation.

The average improvement in packet latency is calculated for both the routing strategies. The average values for sequential implementation is 20.247ns, 14.282ns and 14.285ns for Emesh, ATAC and ORNoC architectures respectively. The average values are 20.1ns for Emesh, 13.907ns for ATAC and 13.91ns for ORNoC for the pure data implementation. 20.037ns for Emesh, 13.926ns for ATAC and 13.87ns for ORNoC are observed to be the average values for mixed implementations. The average variation in latency values against sequential implementation are: 0.73% and 1.04% for pure data parallelism and mixed parallelism implementations respectively for Emesh, 2.6% and 2.5% for pure data parallelism and mixed parallelism implementations respectively for ATAC, 2.6% and 2.9% for pure data parallelism and mixed parallelism implementations respectively for ORNoC. Both the parallelized implementations provide better latency values as compared to the sequential implementation. The mixed parallelism implementation provides better latency values in majority of the cases for all the architectures due to its efficient parallelization implementation as compared to the pure data parallelism implementation.

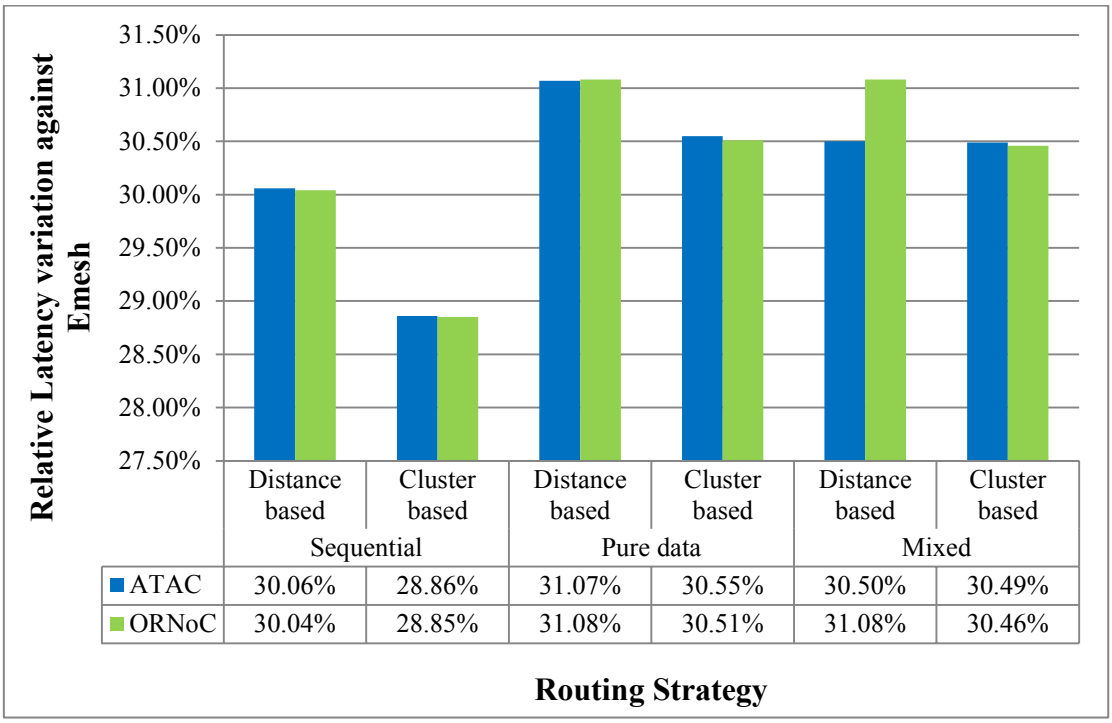


Figure 68: Relative Latency variation for Routing Strategies N = {distance, cluster}

The above plot quantifies the relative variation in latency values of the ONoC architectures against Emesh. The relative packet latency variations for the ATAC architecture are 30.06% for the distance based routing and 28.86% for the cluster based routing strategy in the case of sequential implementation, 31.07% for the distance based routing and 30.55% for the cluster based routing strategy in the case of pure data parallelism and 30.5% for the distance based routing and 30.49% for the cluster based routing strategy in the case of mixed parallelism implementation. The ORNoC architecture shows relative packet latency variation of 30.04% for the distance based routing and 28.85% for the cluster based routing strategy in the case of sequential implementation, 31.08% for the distance based routing and 30.51% for the cluster based routing strategy in the case of pure data parallelism and 31.08% for the distance based routing and 30.46% for the cluster based routing strategy in the case of mixed parallelism implementation. The average improvement in relative variation is calculated for both the

routing strategies. The average latency variation values against Emesh for ATAC is 29.46% and 29.44% for ORNoC in the case of sequential implementation, 30.81% for ATAC and 20.79% for ORNoC in the case of pure data parallelism implementation and 30.49% for ATAC and 30.77% for ORNoC for mixed parallelism implementation. This plot underlines the flexibility of different routing strategies in ONoCs to give improved results compared to the Emesh architecture. The latency values for the Emesh architecture are much greater than that of ATAC and ORNoC as these routing strategies are not applicable for Emesh and the absence of an effective optical communication network.

5.5.2 Results for Contention Delay for varying Routing strategies

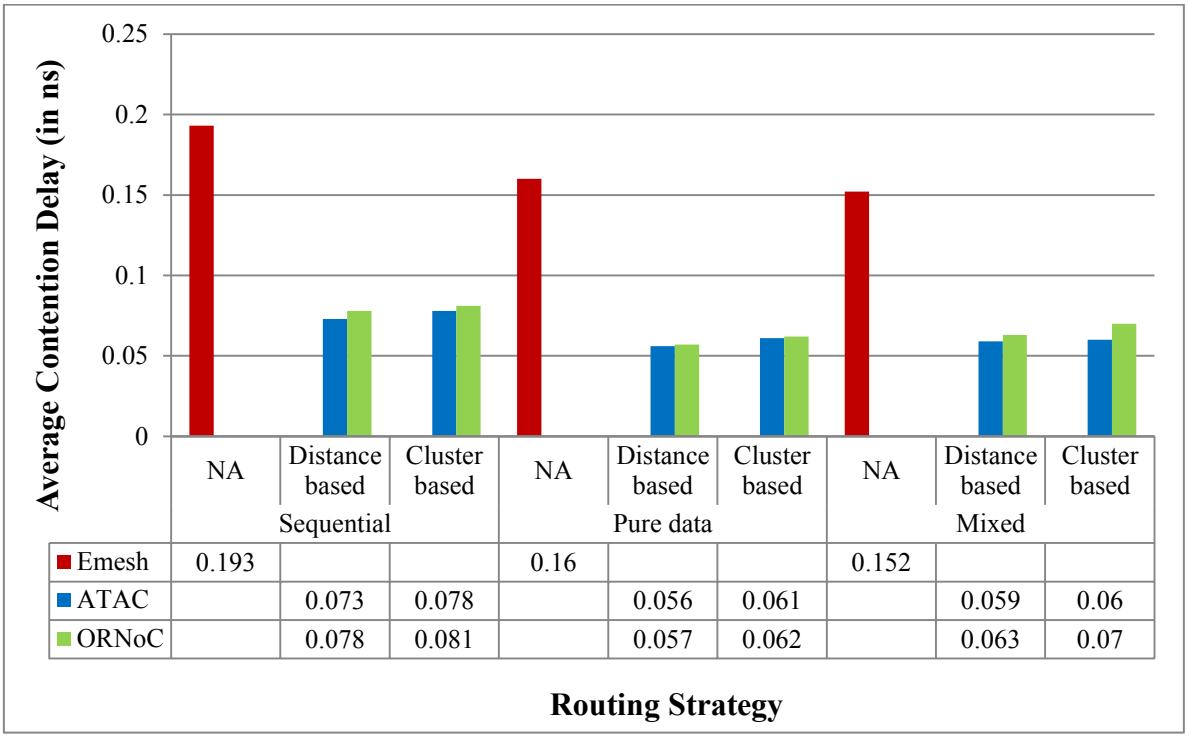


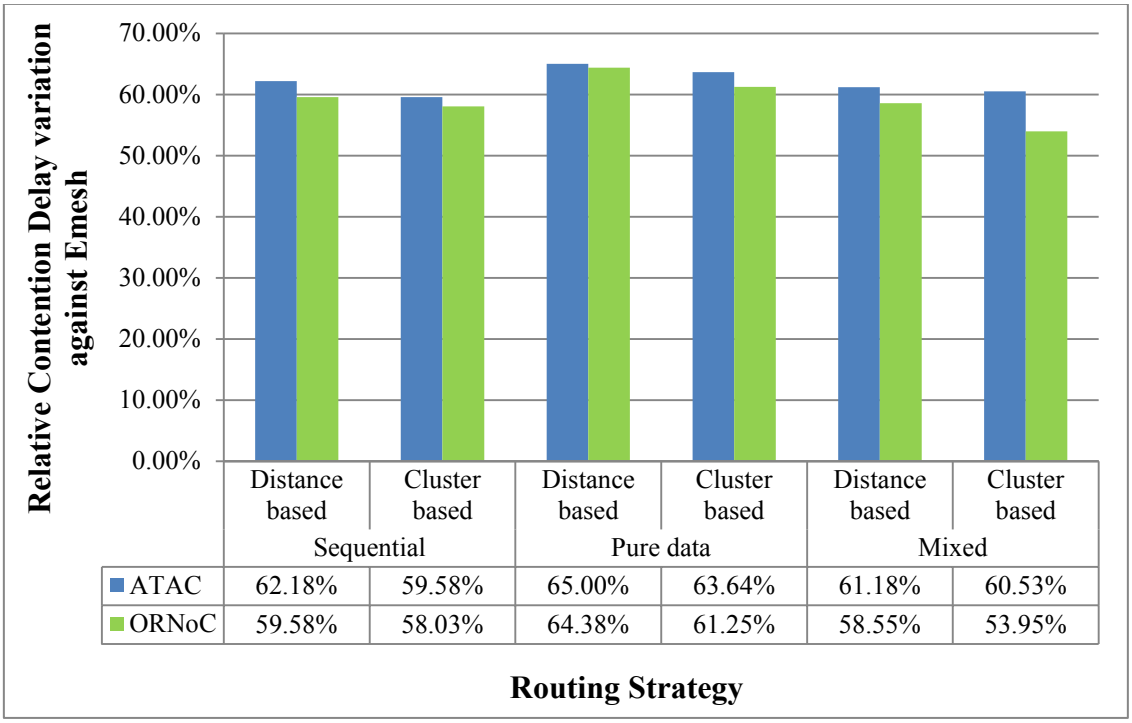
Figure 69: Average Contention Delay for Routing Strategies N = {distance, cluster}

Much similar to the latency values, both the routing strategies have almost similar values for contention delay while the mixed parallelization scheme while the distance based routing

strategy performs slightly better for the other two schemes. Emesh architecture shows the total static energy variations of 0.127J for the sequential implementation, 0.152J for the pure data parallelism implementation and 0.15J for the mixed parallelism implementation. The contention delay values for the ATAC architecture are 0.073ns for the distance based routing and 0.078ns for the cluster based routing strategy in the case of sequential implementation, 0.056ns for the distance based routing and 0.061ns for the cluster based routing strategy in the case of pure data parallelism and 0.059ns for the distance based routing and 0.06ns for the cluster based routing strategy in the case of mixed parallelism implementation. The ORNoC architecture shows contention delay variation of 0.078ns for the distance based routing and 0.081ns for the cluster based routing strategy in the case of sequential implementation, 0.057ns for the distance based routing and 0.062ns for the cluster based routing strategy in the case of pure data parallelism and 0.063ns for the distance based routing and 0.07ns for the cluster based routing strategy in the case of mixed parallelism implementation.

The average improvement in contention delay is calculated for both the routing strategies. The average values of contention delay for sequential implementation are 0.193ns, 0.075ns and 0.079ns for Emesh, ATAC and ORNoC architectures respectively. Pure data parallelism implementation gives average values of 0.16ns for Emesh, 0.058ns for ATAC and 0.059ns for ORNoC architectures. The average values for mixed implementation are 0.152ns, 0.059ns and 0.066ns respectively for Emesh, ATAC and ORNoC architectures. The average variation between the pure data parallelism and mixed parallelism against sequential implementation are 17.10% and 21.24% for Emesh, 22.67% and 21.33% for ATAC and 25.32% and 16.46% for ORNoC respectively. The variation in contention values for the

parallelized implementations against sequential implementation show that the contention delay values are better than that of the sequential implementation due to the efficient parallelization schemes employed. Both the parallelized implementations have values in very close proximity.



	Sequential		Pure data		Mixed	
	Distance based	Cluster based	Distance based	Cluster based	Distance based	Cluster based
ATAC	62.18%	59.58%	65.00%	63.64%	61.18%	60.53%
ORNoC	59.58%	58.03%	64.38%	61.25%	58.55%	53.95%

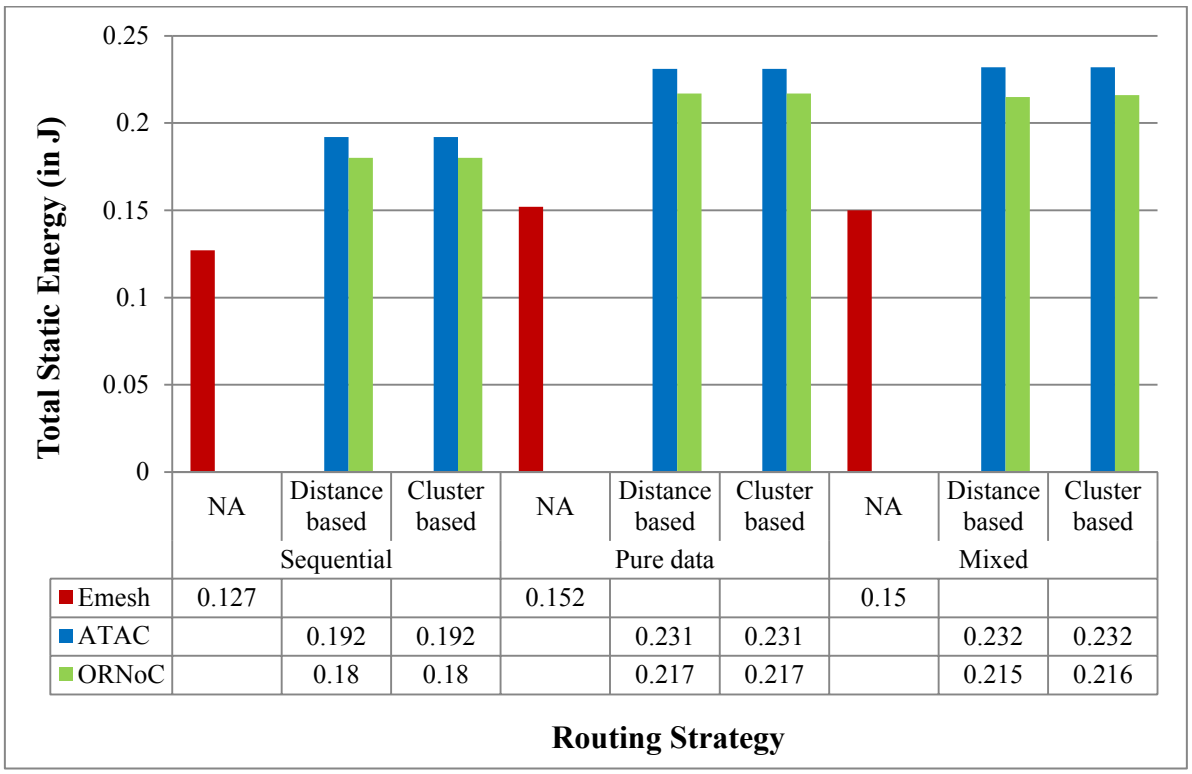
Figure 70: Relative Contention Delay variation for Routing Strategies N = {distance, cluster}

The relative contention delay variations for the ATAC architecture are 62.18% for the distance based routing and 59.58% for the cluster based routing strategy in the case of sequential implementation, 65% for the distance based routing and 63.64% for the cluster based routing strategy in the case of pure data parallelism and 61.18% for the distance based routing and 60.53% for the cluster based routing strategy in the case of mixed parallelism implementation. The ORNoC architecture shows relative contention delay variation of 59.58% for distance based and 58.03% for cluster based routing strategies in the case of

sequential implementation, 64.38% for distance based routing and 61.25% for the cluster based routing strategies in the case of pure data parallelism and 58.55% for the distance based routing and 53.95% for cluster based routing strategy in the case of mixed parallelism implementation.

The average improvement in relative variation is calculated for both the routing strategies. The average improvement in the case of sequential implementation is 60.88% for ATAC and 58.80% for ORNoC against Emesh architecture. The pure data parallelism implementation shows 64.32% improvement for ATAC and 62.81% improvement for ORNoC. The average improvement rate for ATAC is 60.86% and that for ORNoC is 57.24% for the mixed parallelism implementation. The contention delay values are observe to be much better for the ONoC architectures as compared to Emesh, because of different routing strategy in Emesh as well as the absence of an optical communication network.

5.5.3 Results for Static Energy for varying Routing strategies



	Sequential			Pure data			Mixed		
	NA	Distance based	Cluster based	NA	Distance based	Cluster based	NA	Distance based	Cluster based
Emesh	0.127			0.152			0.15		
ATAC		0.192	0.192		0.231	0.231		0.232	0.232
ORNoC		0.18	0.18		0.217	0.217		0.215	0.216

Figure 71: Total Static Energy consumption for Routing Strategies N = {distance, cluster}

Emesh architecture shows the total static energy variations of 0.127J for the sequential implementation, 0.152J for the pure data parallelism implementation and 0.15J for the mixed parallelism implementation. The total static energy values for the ATAC architecture are 0.192J for both the routing strategies in the case of sequential implementation, 0.231J for both the routing strategies in the case of pure data parallelism and 0.232J respectively for both the routing strategies in the case of mixed parallelism implementation. The ORNoC architecture shows relative static energy variation of 0.18J for both the routing strategies in the case of sequential implementation, 0.217J for both the routing strategies in the case of pure data parallelism and 0.215J for the distance based routing and 0.216J for the cluster based routing strategy in the case of mixed parallelism implementation.

The average improvement is calculated for both the routing strategies. The average values for total static energy consumption for the sequential implementation is 0.127J, 0.192J and 0.18J for the Emesh, ATAC and ORNoC architectures respectively. The pure data parallelism implementation shows average values of 0.152J for Emesh, 0.231J for ATAC and 0.217J for ORNoC architectures. Emesh shows average values of 0.15J, ATAC shows 0.232J and ORNoC shows 0.215J for mixed parallelism implementations. The average variation between the pure data parallelism and mixed parallelism against sequential implementation are -19.68% and -18.11% for Emesh, -20.31% and -20.83% for ATAC and -20.56% and -19.44% for ORNoC respectively.

The static energy consumption is greater for the parallelized schemes as compared to the sequential implementation. The number of data packets in the network is observed to be the maximum for the pure data parallelism implementation followed by the mixed parallelism implementation and the least for sequential implementation. Hence, this causes an indirect increase in the data independent energy components, therefore providing maximum energy consumption for pure data parallelism, followed by mixed parallelism and finally by the sequential implementation scheme.

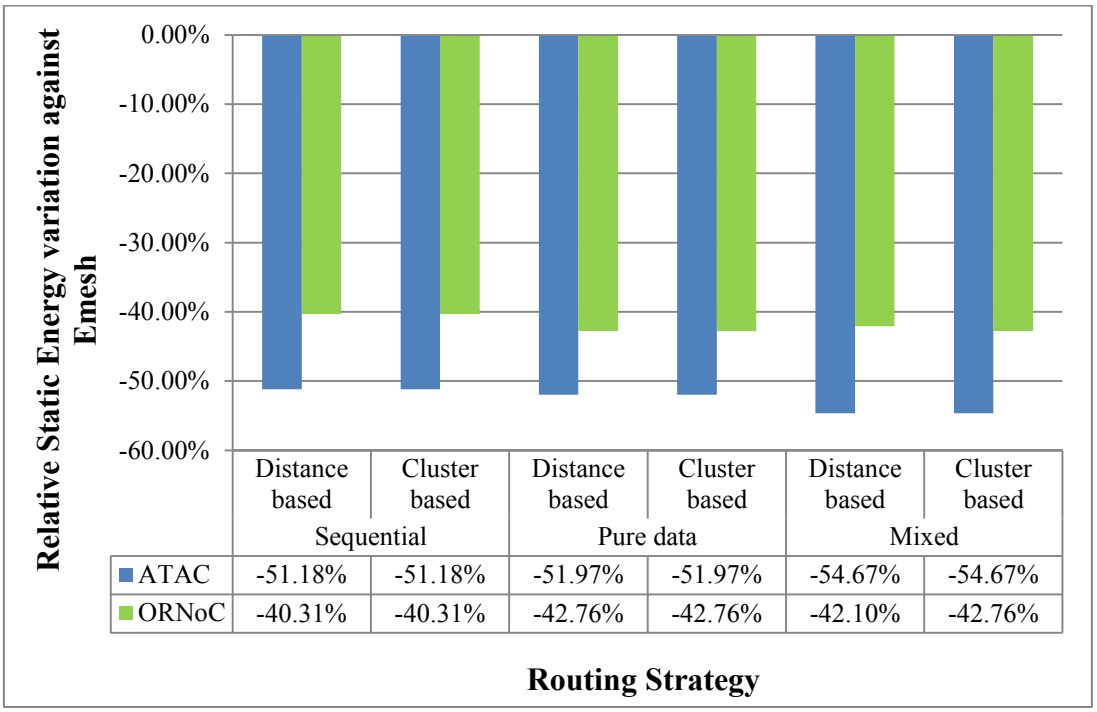


Figure 72: Relative Static Energy consumption for Routing Strategies N = {distance, cluster}

The relative static energy variations for the ATAC architecture are -51.18% for both the routing strategies in the case of sequential implementation, -51.97% for both the routing strategies in the case of pure data parallelism and -54.67% respectively for both the routing strategies in the case of mixed parallelism implementation. The ORNoC architecture shows relative static energy variation of -40.31% for both the routing strategies in the case of sequential implementation, -42.76% for both the routing strategies in the case of pure data parallelism and -42.10% for the distance based routing and -42.76% for the cluster based routing strategy in the case of mixed parallelism implementation.

The average improvement in the relative variation is calculated for both the routing strategies. The average degeneration rate is observed to be -51.18% for ATAC and -40.31% for ORNoC for the sequential implementation, -51.97% and -42.76% for ATAC and ORNoC

respectively for pure data implementation, -54.67% for ATAC and -42.43% for ORNoC the case of mixed parallelism implementation. Optical network architectures have greater static energy consumption than Emesh in static energy consumption as there are no optical components in the network [54]. Among the optical networks, ORNoC architecture provides better results as the number of photonic components that account for static energy which requires electrical circuitry is lesser than in ATAC due to wavelength reuse [54].

5.5.4 Results for Dynamic Energy for varying Routing strategies

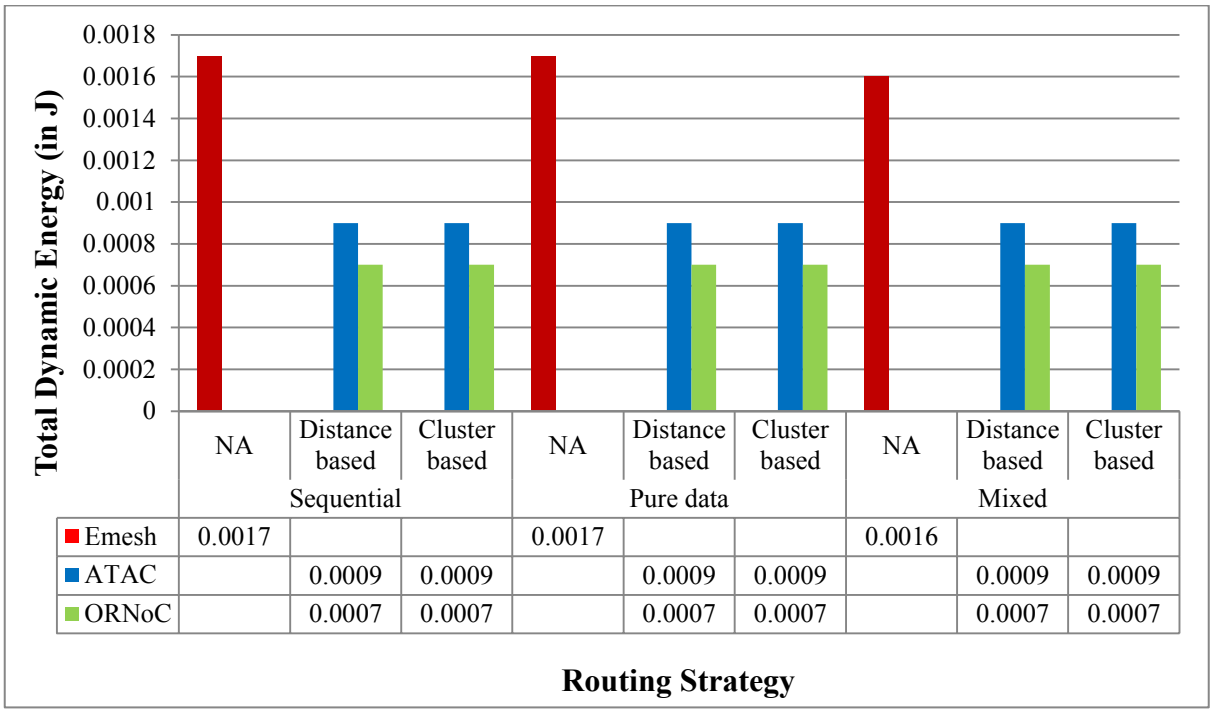


Figure 73: Total Dynamic Energy consumption for different Routing Strategies N = {distance, cluster}

The dynamic energy consumption values are observed to be the same for all the cases in the ONoC architectures for both distance based and cluster based routing schemes for all the implementation schemes. The actual and average dynamic energy consumption values are 0.0009J and 0.0007J for ATAC and ORNoC respectively for all the implementations, while

the Emesh architecture has average values of 0.0017J for both sequential and pure data parallelism implementation and 0.0016J for mixed parallelism implementation. The average variations are calculated for both the routing strategies considered. The different implementations do not show much difference as the application works primarily with the same workload.

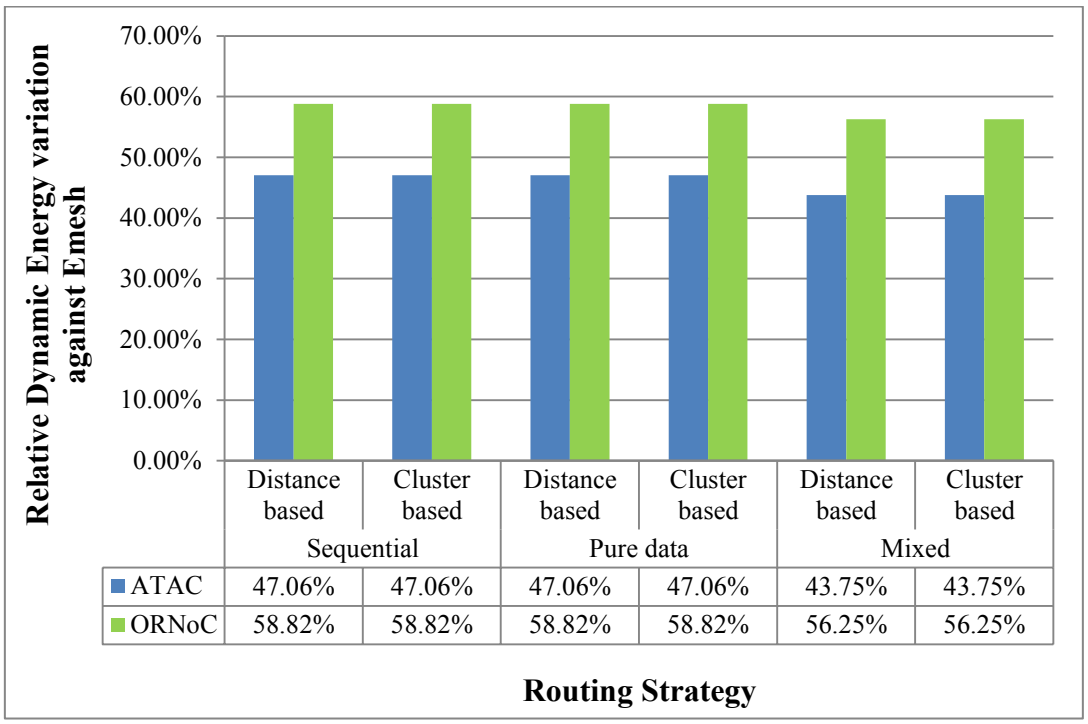


Figure 74: Relative Dynamic Energy consumption variation for Routing Strategies N = {distance, cluster}

The relative dynamic energy variations remain the same for each of the architectures when there is a different routing strategy. The relative variation is the same for both the sequential and pure data implementations, while it decreases slightly for the mixed data parallelization scheme. The relative total energy variations for the ATAC architecture are 47.06% for both the routing strategies in the case of sequential implementation, 47.06% for both the routing strategies in the case of pure data parallelism and 43.75% respectively for both the routing

strategies in the case of mixed parallelism implementation. The ORNoC architecture shows relative variation total energy variation of 58.82% for both the routing strategies in the case of sequential implementation, 58.82% for both the routing strategies in the case of pure data parallelism and 56.25% for both the routing strategies in the case of mixed parallelism implementation.

The average relative improvement is calculated for both the routing strategies. The average improvement rate is observed to be 47.06% for ATAC and 58.82% for ORNoC for the sequential implementation, 47.06% and 58.82% for ATAC and ORNoC respectively for pure data implementation, 43.75% for ATAC and 56.25% for ORNoC in the case of mixed parallelism implementation. The Emesh architecture has almost similar values for all the implementations, but shows higher energy consumption as compared to the ONoC architectures as they have larger number of electrical links which contribute to higher dynamic energy consumption [54].

5.5.5 Results for Total Energy for varying Routing strategies

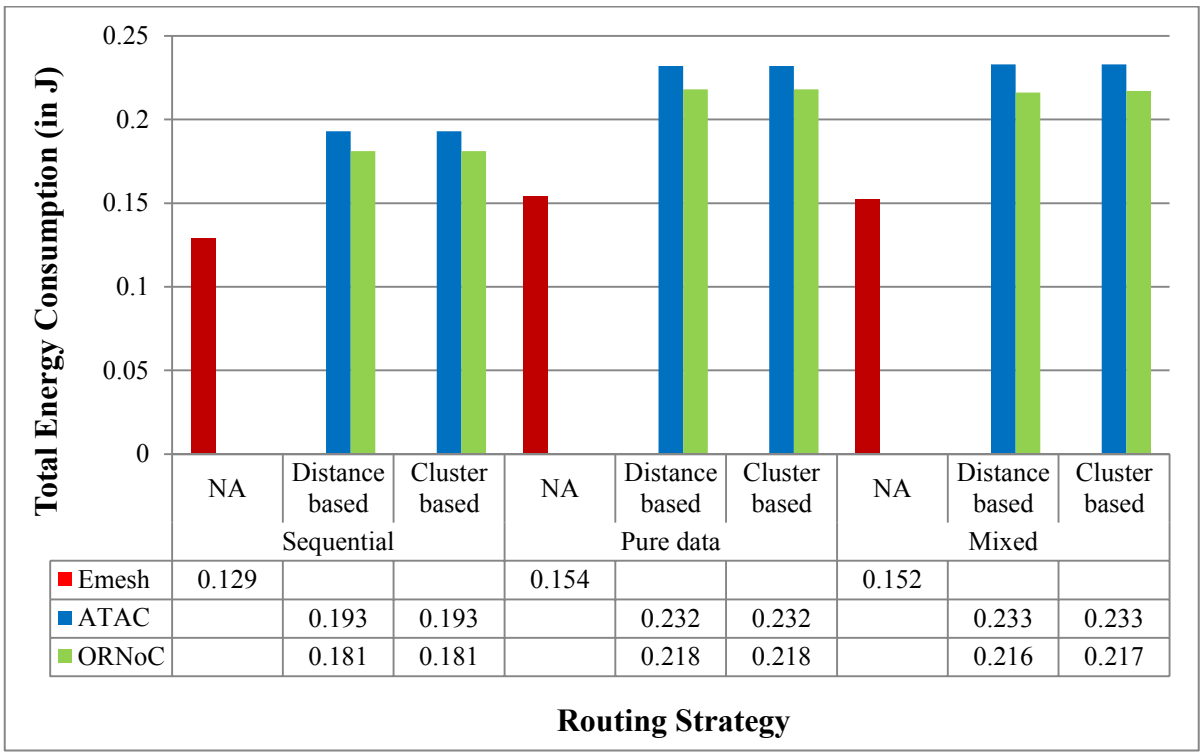


Figure 75: Total Energy consumption for Routing Strategies N = {distance, cluster}

Majority of the total energy consumption is constituted by the static energy consumption as dynamic energy values form only a very small fraction of the total energy consumption. Hence, the trend is observed to be similar to that of the static energy consumption. The total energy consumption for Emesh is 0.129J, 0.154J and 0.152J in the case of sequential, ure data and mixed parallelism implementations respectively. The total energy values for the ATAC architecture are 0.193J for both the routing strategies in the case of sequential implementation, 0.232J for both the routing strategies in the case of pure data parallelism and 0.233J for both the routing strategies in the case of mixed parallelism implementation. The ORNoC architecture shows total energy variation of 0.181J for both the routing strategies in the case of sequential implementation, 0.218J for both the routing strategies in

the case of pure data parallelism and 0.216J for the distance based routing and 0.217J for the cluster based routing strategies in the case of mixed parallelism implementation. The average variations are calculated for both the routing strategies considered. The average values for total energy consumption in the case of sequential implementation is 0.129J are Emesh, 0.193J for ATAC and 0.181J for ORNoC, while pure data parallelism implementation has 0.154J for Emesh, 0.232J for ATAC and 0.218J for ORNoC and mixed implementation gives the results of 0.152J, 0.233J and 0.216J for Emesh, ATAC and ORNoC architectures respectively. The pure data implementation shows an average variation of -21.26% while the mixed implementation shows an average variation of -19.68% with respect to the sequential implementation for the Emesh architecture, while the average variation against sequential implementation is -50.65% and -120.13%for pure data parallelism and mixed parallelism implementations respectively for the ATAC architecture, while the variation rates are -41.56% for pure data parallelism and -42.10% for the ORNoC architecture. All the implementation schemes follow the same trend as that in the case of static energy consumption.

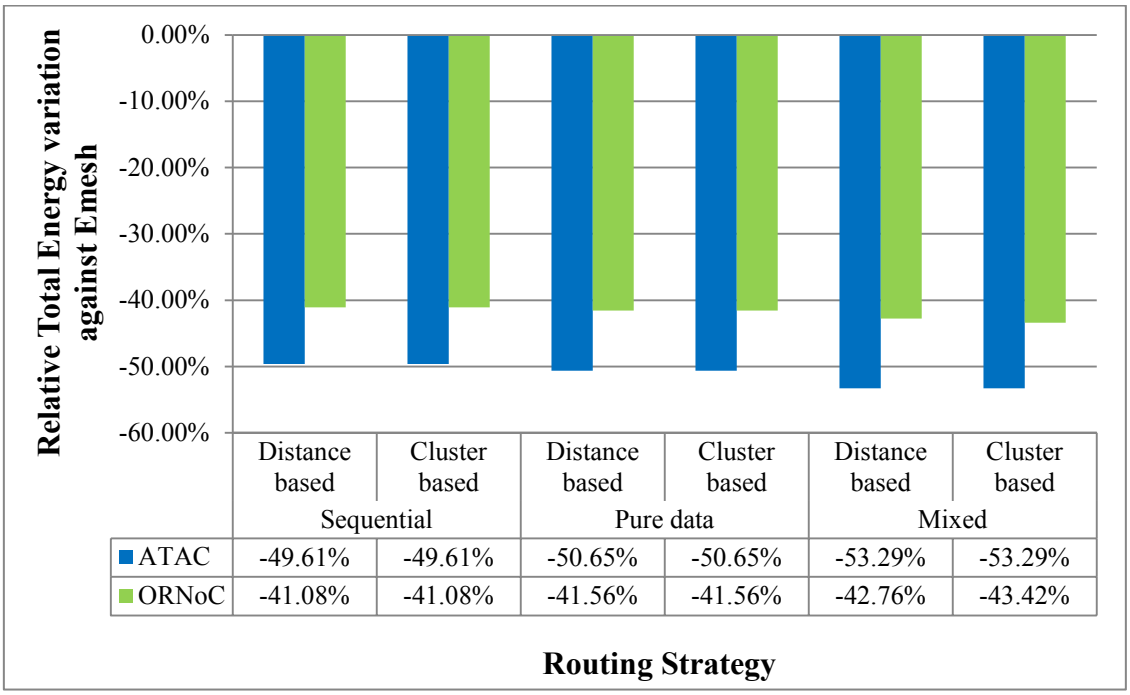


Figure 76: Relative Total Energy variation for Routing Strategies N = {distance, cluster}

The values remain constant for the different implementation schemes for each of the architectures. The relative total energy variations for the ATAC architecture are -49.61% for both the routing strategies in the case of sequential implementation, -50.65% for both the routing strategies in the case of pure data parallelism and -53.29% respectively for both the routing strategies in the case of mixed parallelism implementation. The ORNoC architecture shows relative variation total energy variation of -41.08% for both the routing strategies in the case of sequential implementation, -41.56% for both the routing strategies in the case of pure data parallelism and -42.76% for both the routing strategies in the case of mixed parallelism implementation.

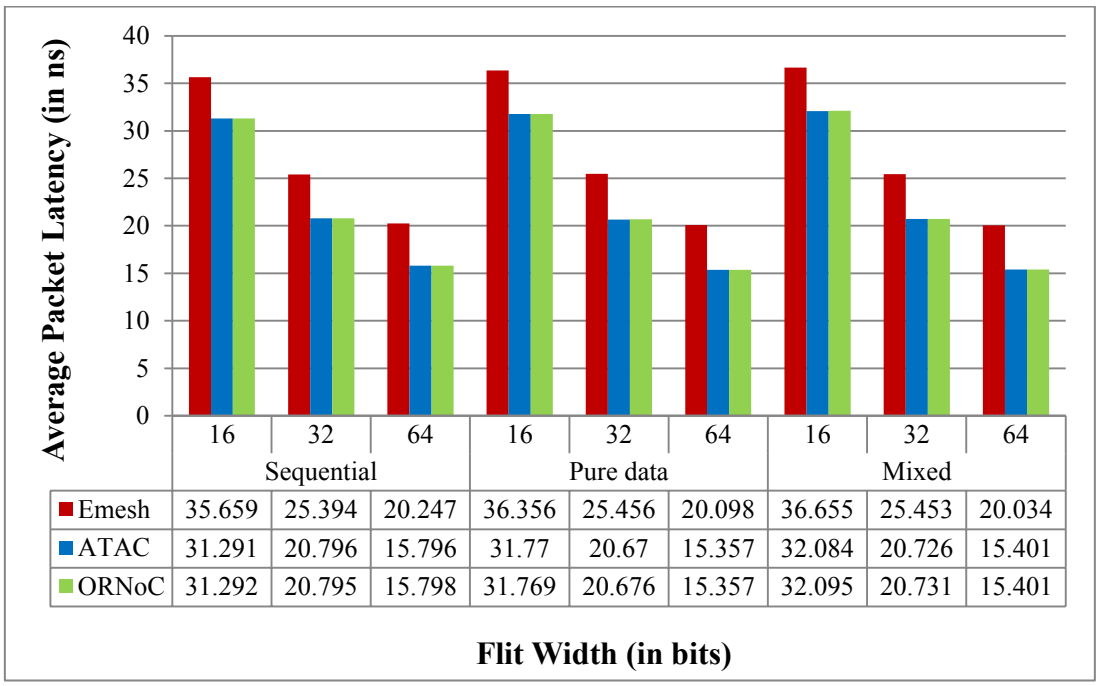
The average relative variations are calculated for both the routing strategies considered for all the implementations. The average degradation rates are observed to be -49.61% and -41.08% for ATAC and ORNoC respectively in the case of sequential implementation, -50.65% for ATAC and -41.56% for ORNoC in the case of pure data parallelism

implementation, -121.71% and -43.09% for the mixed parallelism implementation. The degradation rate of ATAC architecture is observed to be much larger as compared to the ORNoC architecture.

5.6 Analysis with varying Flit Width Values

Packets of data are divided into smaller constituent units known as flits. A packet usually consists of a head flit, which contains information regarding the destination, one or more body flits having the actual payload and a tail flit indicating the end of transaction. The variation in flit width would alter the size of each data packet flowing through the network to actually process the same amount of data. Hence, the decrease in flit width will have more packets flowing through the network but smaller one time transactions, while increase in flit width would average lesser transactions with larger packets. This experiment uses a 1024 pixel square image for which is executed on 64 cores with the cluster size of 16. This image provides moderate workload and uses 4 clusters, which can portray the variation in results for this experiment clearly. The rest of the configurations are kept as default and mentioned in Table 4. The detailed configuration is provided in Appendix Table 6 .

5.6.1 Results for Packet Latency for varying Flit Width Values



	16	32	64	16	32	64	16	32	64
	Sequential			Pure data			Mixed		
Emesh	35.659	25.394	20.247	36.356	25.456	20.098	36.655	25.453	20.034
ATAC	31.291	20.796	15.796	31.77	20.67	15.357	32.084	20.726	15.401
ORNoC	31.292	20.795	15.798	31.769	20.676	15.357	32.095	20.731	15.401

Figure 77: Average Packet Latency for Flit Widths N = {16,32,64}

The average packet latency values decrease with increasing flit widths. The increase in flit width causes increase in the size of each packet which in turn decreases the number of packets required to process the same amount of data. For the Emesh architecture, the packet latency increases from 35.659ns when the flit width is 16 bits to 20.247ns for 64 bits in the case of sequential implementation, while it is between 36.356ns and 20.098ns for pure data parallelism implementation for 16 bits and 64 bits respectively and the values are between 36.655ns for 16 bits and 20.034ns for the 64 bits for the mixed implementation. The ATAC architecture shows the packet latency values from 31.291ns when the flit width is 16 bits and ranges to15.796ns for 64 bits in the case of sequential implementation, while it is between 31.77ns and 15.357ns for pure data parallelism implementation for 16 bits and 64 bits respectively and the values are between 32.084ns for 64 bits and 15.401ns for 64 bits for the

mixed implementation. For the ORNoC architecture, the packet latency values increase from 31.292ns when the flit width is 16 bits to 15.798ns for 64 bits in the case of sequential implementation, while it is between 31.769ns and ranges to 15.357ns for pure data parallelism implementation for 16 bits and 64 bits respectively and the values are between 32.095ns for 16 bits and 15.401ns for 64 bits in the case of mixed parallelism implementation.

The packet latency output gives average values for all the flit width data points for all implementations that are 27.1ns for Emesh architecture while it shows 22.63ns for both ATAC and ORNoC architectures in the case of sequential implementation. The average values for all the flit width values for pure data parallelism implementation is 27.3ns for Emesh, 22.6ns for both ATAC and ORNoC architectures, while the mixed implementation gives the results of 27.38ns for Emesh and 22.74ns for both ATAC and ORNoC architectures. The pure data implementation shows an average variation of -0.74% while the mixed implementation shows an average variation of -1.03% with respect to the sequential implementation for the Emesh architecture, while the average variation against sequential implementation is 0.13% and -0.49% for pure data parallelism and mixed parallelism implementations respectively for both the ATAC and ORNoC architectures. For the different implementations, the parallelized implementations perform better for higher flit widths because for the lower flit widths, the number of packets increase and the three implementations use almost the same number of packets in the network, which provides very little variation in the latency values among the three implementations.

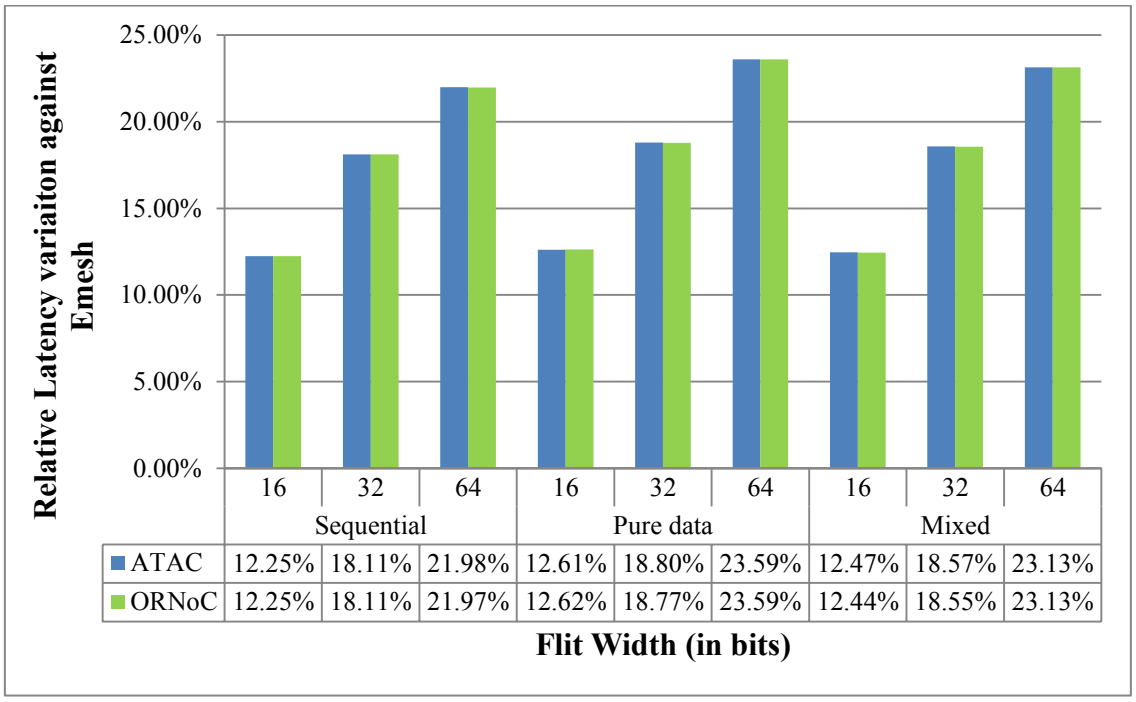
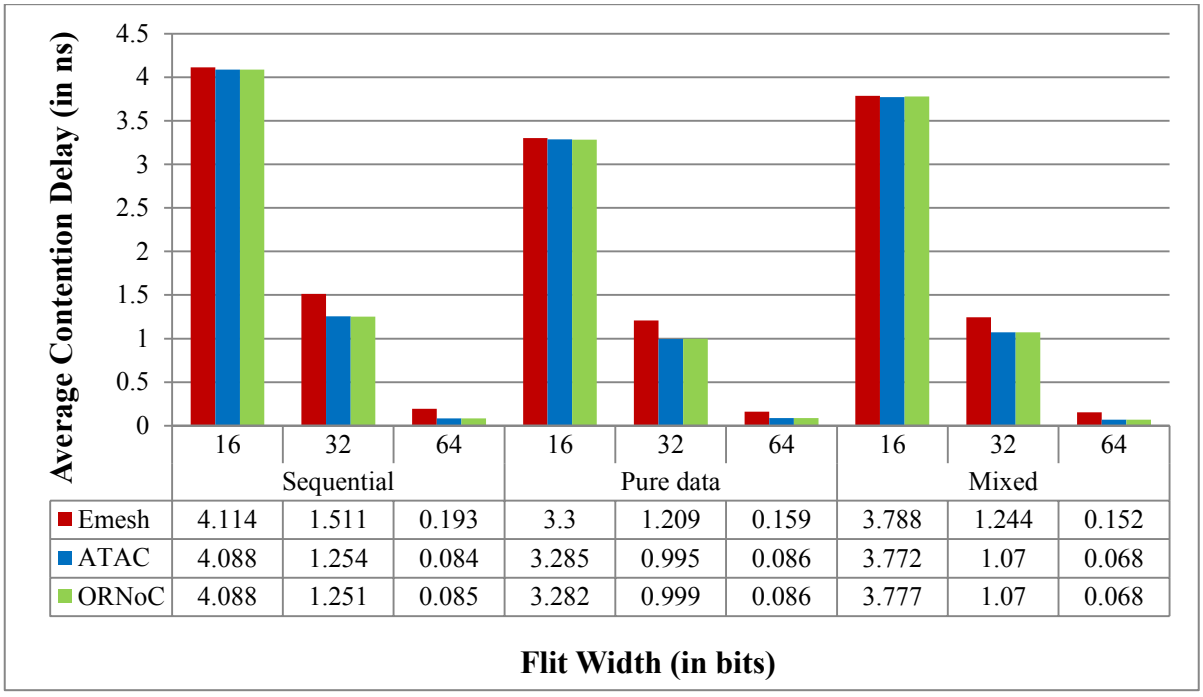


Figure 78: Relative Latency variation for Flit Widths N = {16,32,64}

The above plot quantifies the improvement in the packet latency values of both ONoC architectures with respect to the Emesh architecture. The relative improvement in packet latency in the case of ATAC architecture is 12.25% when flit width is 16 bits and increases to 21.98% in the case when 64 bits for sequential implementation, goes from 12.61% in the case when the flit width is 16 bits to 23.59% for flit width of 64 bits in the case of pure data parallelism implementation and 12.47% and 23.13% for 16 bit and 64 bit flit widths respectively in the case of mixed parallelism implementation. The ORNoC architecture has relative packet latency values of 12.25% when flit width is 16 bits and increases to 21.97% in the case when 64 bits for sequential implementation, goes from 12.62% in the case when the flit width is 16 bits to 23.59% for flit width of 64 bits in the case of pure data parallelism implementation and 12.44% and 23.13% for 16 bit and 64 bit flit widths respectively in the case of mixed parallelism implementation.

The average relative variations are calculated for all the flit width data points for all the implementations. The average improvement rates against Emesh is 17.45% for ATAC and 17.44% for ORNoC architectures respectively in the case of sequential implementation, for pure data parallelism implementation it is 18.33% for both ATAC and ORNoC architectures and 18.06% for ATAC and 18.04% for ORNoC architectures in the case of mixed parallelization. Hence, it is observed that the latency values show the best results when flit width is 64 bits. The graph indicates that the ONoC architectures have better latency values than Emesh as it uses optical network also for communication that reduces the packet latency.

5.6.2 Results for Contention Delay for varying Flit Width Values



	Sequential			Pure data			Mixed		
	16	32	64	16	32	64	16	32	64
Emesh	4.114	1.511	0.193	3.3	1.209	0.159	3.788	1.244	0.152
ATAC	4.088	1.254	0.084	3.285	0.995	0.086	3.772	1.07	0.068
ORNoC	4.088	1.251	0.085	3.282	0.999	0.086	3.777	1.07	0.068

Figure 79: Average Contention Delay for Flit Widths N = {16, 32, 64}

The results show average contention delay decreases considerably with the increase in flit width values for all the architectures and all the parallelization schemes. The flit width

increment reduces the number of packets to be processed for the workload and hence, considerably improves contention delay For the Emesh architecture, the contention delay increases from 4.114ns when the flit width. is 16 bits to 0.193ns for 64 bits in the case of sequential implementation, while it is between 3.3ns and 0.159ns for pure data parallelism implementation for 16 bits and 64 bits respectively and the values are between 3.788ns for 16 bits and 0.152ns for the 64 bits for the mixed implementation. The ATAC architecture shows the contention delay values from 4.088ns when the flit width is 16 bits and ranges to 0.084ns for 64 bits in the case of sequential implementation, while it is between 3.285ns and 0.086ns for pure data parallelism implementation for 16 bits and 64 bits respectively and the values are between 3.772ns for 64 bits and 0.068ns for 64 bits for the mixed implementation. For the ORNoC architecture, the contention delay values increase from 4.088ns when the flit width is 16 bits to 0.085ns for 64 bits in the case of sequential implementation, while it is between 3.282ns and ranges to 0.85ns for pure data parallelism implementation for 16 bits and 64 bits respectively and the values are between 3.777ns for 16 bits and 0.068ns for 64 bits in the case of mixed parallelism implementation.

The average variations are calculated for all the flit width data points for all the implementations. The average values of contention delay for sequential implementation are 1.94ns, 1.809ns and 1.808ns for Emesh, ATAC and ORNoC architectures respectively. Pure data parallelism implementation gives average values of 1.56ns for Emesh, 1.45ns for ATAC and 1.456ns for ORNoC architectures. The average values for mixed implementation are 1.73ns, 1.64ns and 1.638ns respectively for Emesh, ATAC and ORNoC architectures. The average variation between the pure data parallelism and mixed parallelism against sequential implementation are 19.59% and 10.82% for Emesh, 19.84% and 9.34% for ATAC and

19.47% and 9.40% for ORNoC respectively. The sequential implementation is observed to have increased contention delay values as compared to both the parallelized implementations, as the parallelized implementations utilize effective distribution of data using threads which reduces the contention delay. Both the parallelized implementations have little variations for contention delay.

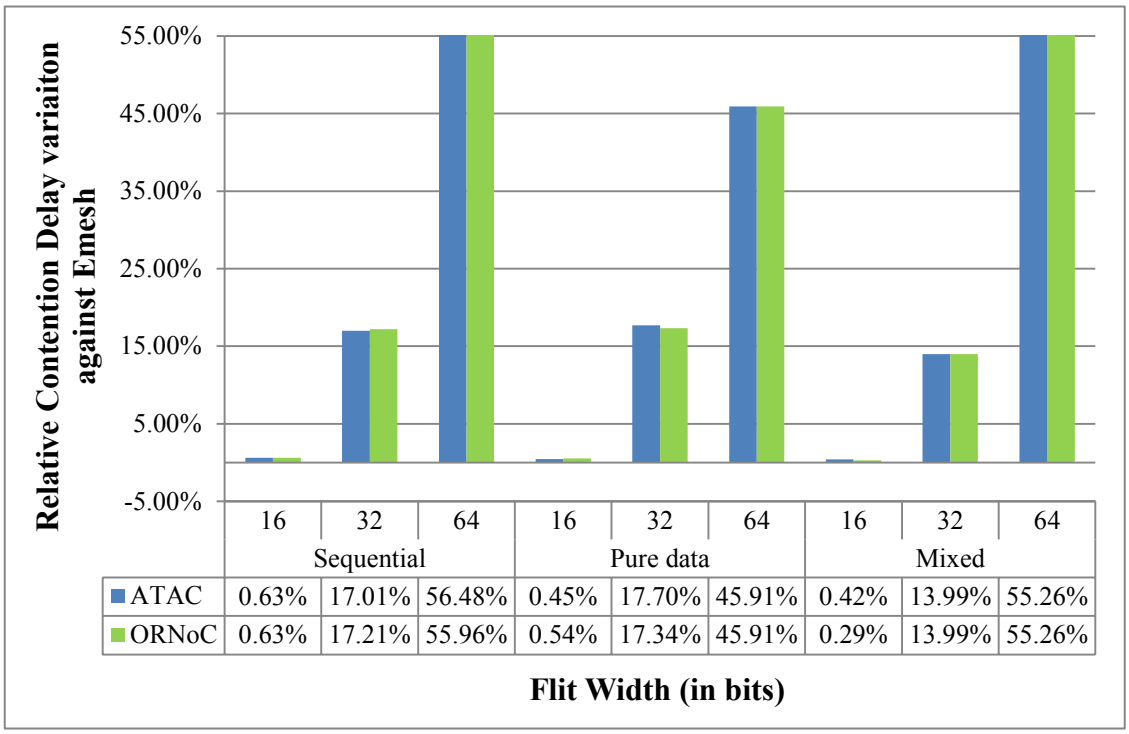


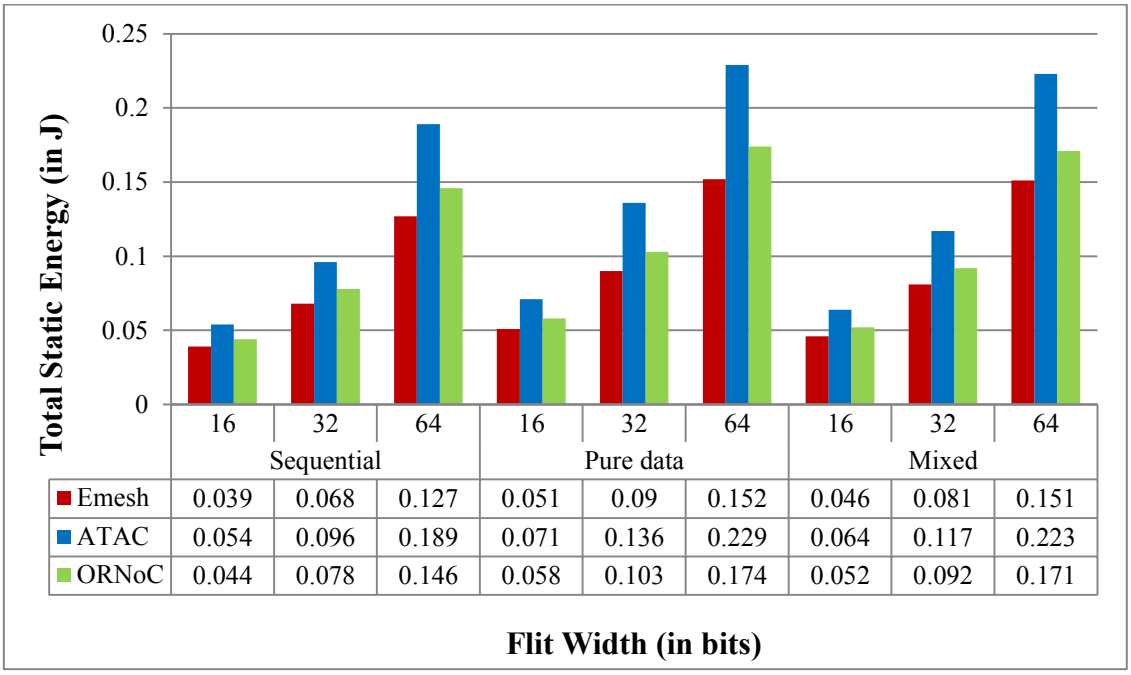
Figure 80: Relative Contention Delay variation for Flit Widths N = {16, 32, 64}

The above plot indicates the improvement in contention delay for the ONoC architectures against Emesh architecture. The relative improvement in contention delay in the case of ATAC architecture is 0.63% when flit width is 16 bits and increases to 56.48% in the case when 64 bits for sequential implementation, goes from 0.45% in the case when the flit width is 16 bits to 45.91% for flit width of 64 bits in the case of pure data parallelism implementation and 0.42% and 55.26% for 16 bits and 64 bit flit widths respectively in the case of mixed parallelism implementation. The ORNoC architecture has relative contention

delay values of 0.63% when flit width is 16 bits and increases to 55.96% in the case when 64 bits for sequential implementation, goes from 0.54% in the case when the flit width is 16 bits to 45.91% for flit width of 64 bits in the case of pure data parallelism implementation and 0.29% and 55.26% for 16 bit and 64 bit flit widths respectively in the case of mixed parallelism implementation.

The average relative variations are calculated for all the flit width data points for all the implementations. The average improvement rates with respect to Emesh are 24.71% for ATAC and 24.6% for ORNoC architectures in the case of sequential implementation. The pure data parallelism implementation shows average of 21.35% improvement for ATAC and 21.26% improvement for ORNoC architectures, while ATAC shows 23.22% and ORNoC shows 23.18% improvement the case of mixed parallelization. Hence, it is observed that the contention delay improvement is the best when flit width is 64 bits. The ONoC architectures provide better contention delay values as compared to Emesh architecture due to the presence of optical networks for communication in the network.

5.6.3 Results for Static Energy for varying Flit Width Values



	16	32	64	16	32	64	16	32	64
	Sequential			Pure data			Mixed		
Emesh	0.039	0.068	0.127	0.051	0.09	0.152	0.046	0.081	0.151
ATAC	0.054	0.096	0.189	0.071	0.136	0.229	0.064	0.117	0.223
ORNoC	0.044	0.078	0.146	0.058	0.103	0.174	0.052	0.092	0.171

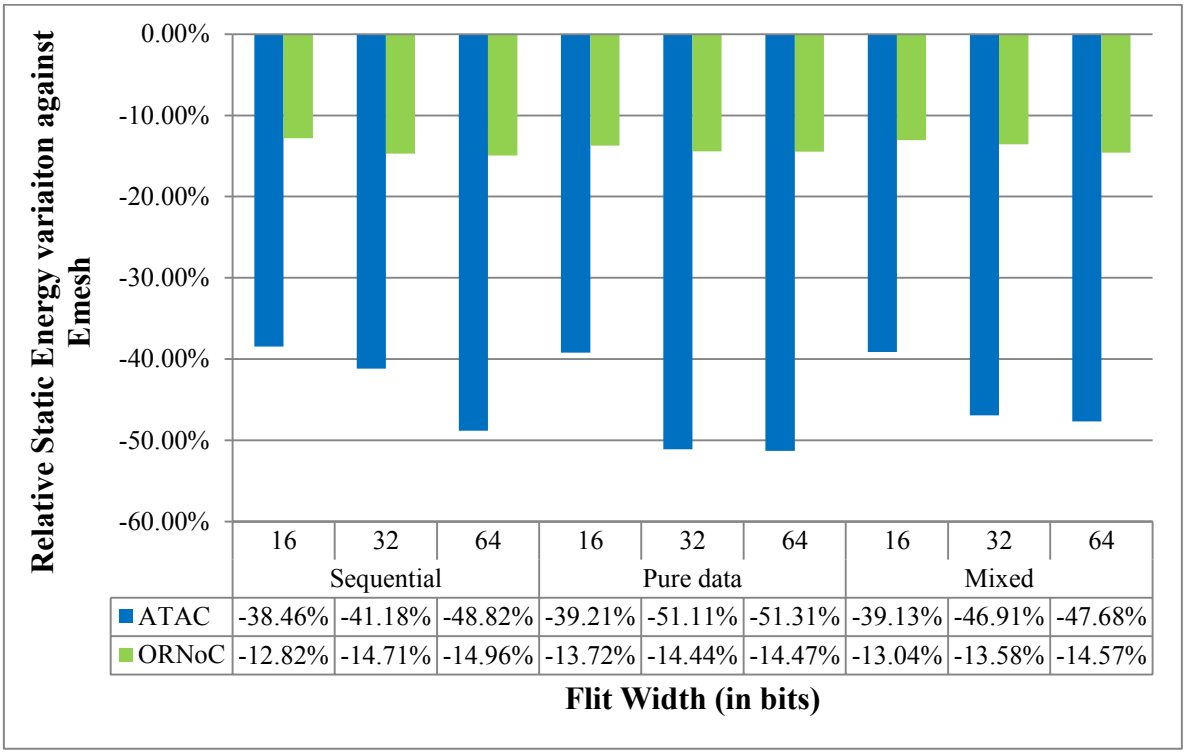
Figure 81: Total Static Energy for Flit Widths N = {16, 32, 64}

Static energy consumption values are lesser for Emesh architecture when compared to the ONoC architectures as it has no optical components. For the Emesh architecture, the static energy increases from 0.039J when the flit width is 16 bits to 0.127J for 64 bits in the case of sequential implementation, while it is between 0.051J and 0.152J for pure data parallelism implementation for 16 bits and 64 bits respectively and the values are between 0.046J for 16 bits and 0.151J for the 64 bits for the mixed implementation. The ATAC architecture shows the static energy values from 0.054J when the flit width is 16 bits and ranges to 0.189J for 64 bits in the case of sequential implementation, while it is between 0.071J and 0.229J for pure data parallelism implementation for 16 bits and 64 bits respectively and the values are between 0.064J for 64 bits and 0.223J for 64 bits for the mixed implementation. For the ORNoC architecture, the static energy values increase from 0.044J when the flit width is 16

bits to 0.146J for 64 bits in the case of sequential implementation, while it is between 0.064J and ranges to 0.223J for pure data parallelism implementation for 16 bits and 64 bits respectively and the values are between 0.052J for 16 bits and 0.171J for 64 bits in the case of mixed parallelism implementation.

The average values of static energy consumption for all the flit widths for all the implementations are 0.078J for Emesh, 0.113J for ATAC and 0.089J for ORNoC architectures in the case of sequential implementation. The average values for all the flit widths for pure data parallelism implementation is 0.098J, 0.145J and 0.112J for Emesh, ATAC and ORNoC architectures respectively, while the mixed implementation gives 0.093J for Emesh and 0.135J for ATAC and 0.105J for ORNoC architectures as the average values for static energy consumption. The pure data implementation shows an average variation of -25.64% while the mixed implementation shows an average variation of -19.23% with respect to the sequential implementation for the Emesh architecture, while the average variation against sequential implementation is -28.32% and -19.47% for pure data parallelism and mixed parallelism implementations respectively for ATAC architecture and the variation is -25.84% for pure data parallelism implementation and -17.98% for the mixed parallelism implementation for the ORNoC architecture. The static energy consumption is greater for the parallelized schemes as compared to the sequential implementation. The number of data packets in the network is observed to be the maximum for the pure data parallelism implementation followed by the mixed parallelism implementation and the least for sequential implementation. Hence, this causes an indirect increase in the data independent energy components, therefore providing maximum energy consumption for pure data

parallelism, followed by mixed parallelism and finally by the sequential implementation scheme.



	16	32	64	16	32	64	16	32	64
	Sequential			Pure data			Mixed		
ATAC	-38.46%	-41.18%	-48.82%	-39.21%	-51.11%	-51.31%	-39.13%	-46.91%	-47.68%
ORNoC	-12.82%	-14.71%	-14.96%	-13.72%	-14.44%	-14.47%	-13.04%	-13.58%	-14.57%

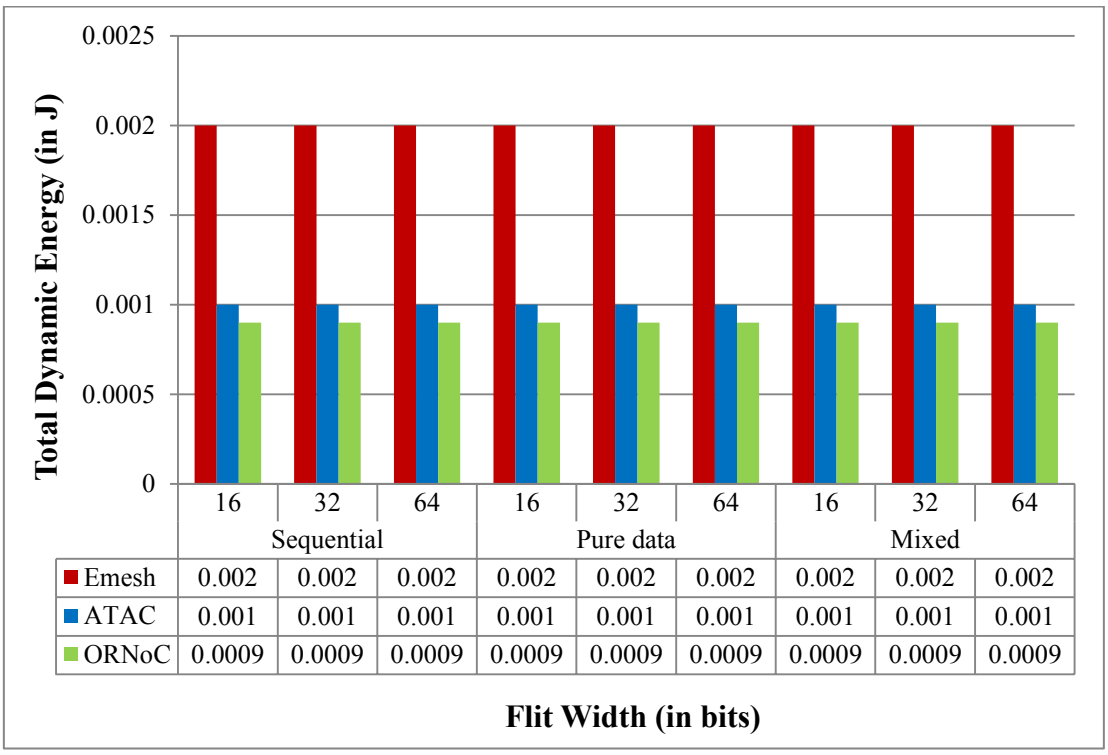
Figure 82: Relative Static Energy variation for Flit Widths N = {16, 32, 64}

The relative static energy variation graphs show that the degradation in static energy is higher for ATAC architecture as compared to the ORNoC architecture with different flit widths. The degradation relative to Emesh for both the architectures is observed to increase with increasing flit widths for almost all the cases. The relative improvement in contention delay in the case of ATAC architecture is -38.46% when flit width is 16 bits and increases to -48.82% in the case when 64 bits for sequential implementation, goes from -39.21% in the case when the flit width is 16 bits to -51.31% for flit width of 64 bits in the case of pure data parallelism implementation and -39.13% and -47.68% for 16 bit and 64 bit flit widths respectively in the case of mixed parallelism implementation. The ORNoC architecture has relative contention delay values of -12.82% when flit width is 16 bits and increases to -

14.96% in the case when 64 bits for sequential implementation, goes from -13.72% in the case when the flit width is 16 bits to -14.47% for flit width of 64 bits in the case of pure data parallelism implementation and -13.04% and -14.57% for 16 bit and 64 bit flit widths respectively in the case of mixed parallelism implementation.

The average relative variations are calculated for all the flit width data points considered. The ATAC architecture shows average degeneration rates of -42.82% and ORNoC shows degeneration rates of -14.16% for sequential implementation for all the flit widths considered. The average decline rates for all the flit widths in the case of pure data parallelism implementation are -47.21% for ATAC and -14.21% for ORNoC architectures. The mixed parallelism implementation shows average decline rates of -44.57% and -13.73% for ATAC and ORNoC architectures respectively for all the cache line sizes. Optical network architectures are outperformed by Emesh in static energy consumption as it is lesser without any optical components in the network [54]. Among the optical networks, ORNoC architecture provides better results as the number of photonic components that require electrical circuitry support is lesser than in ATAC due to wavelength reuse [54].

5.6.4 Results for Dynamic Energy for varying Flit Width Values



	16	32	64	16	32	64	16	32	64
	Sequential			Pure data			Mixed		
Emesh	0.002	0.002	0.002	0.002	0.002	0.002	0.002	0.002	0.002
ATAC	0.001	0.001	0.001	0.001	0.001	0.001	0.001	0.001	0.001
ORNoC	0.0009	0.0009	0.0009	0.0009	0.0009	0.0009	0.0009	0.0009	0.0009

Figure 83: Total Dynamic Energy for Flit Widths N = {16,32,64}

The dynamic energy values are observed to be constant for all the different implementation schemes for each of the architectures with increase in flit width. The actual and average values for all flit widths are equal for each of the architectures and they are 0.002J for Emesh, 0.001J for ATAC and 0.0009J for ORNoC for all the three implementations. As clear from the diagram the ORNoC has better dynamic energy consumption values as compared to ATAC, which in turn has better values compared to Emesh. Emesh architecture has more number of electrical links that contribute to the dynamic energy and hence, it is greater for Emesh architecture as compared to ONoC architectures [54]. ORNoC architecture uses lesser number of wavelengths and waveguides which in turn lead to decreased number of modulators and receivers that contribute to data dependent energy consumption and

hence, has better dynamic energy consumption as compared to ATAC [54]. The different implementation schemes have similar dynamic energy consumption as they work on the same amount of data, which constitutes the dynamic energy consumption.

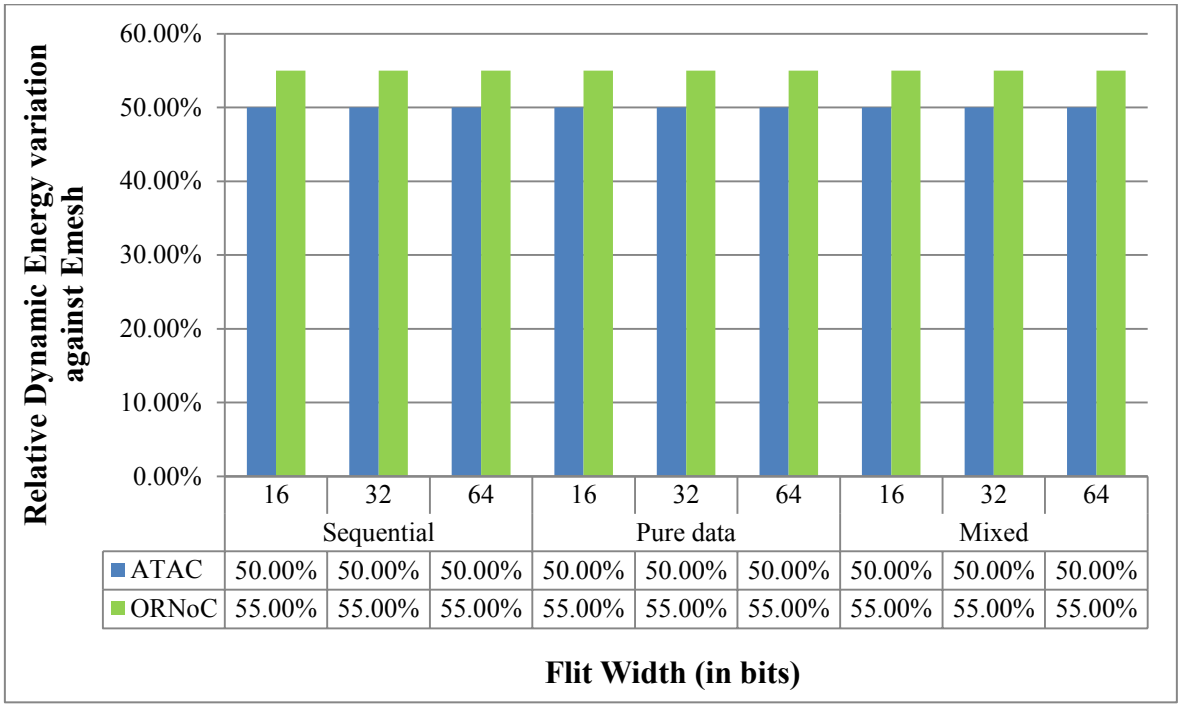
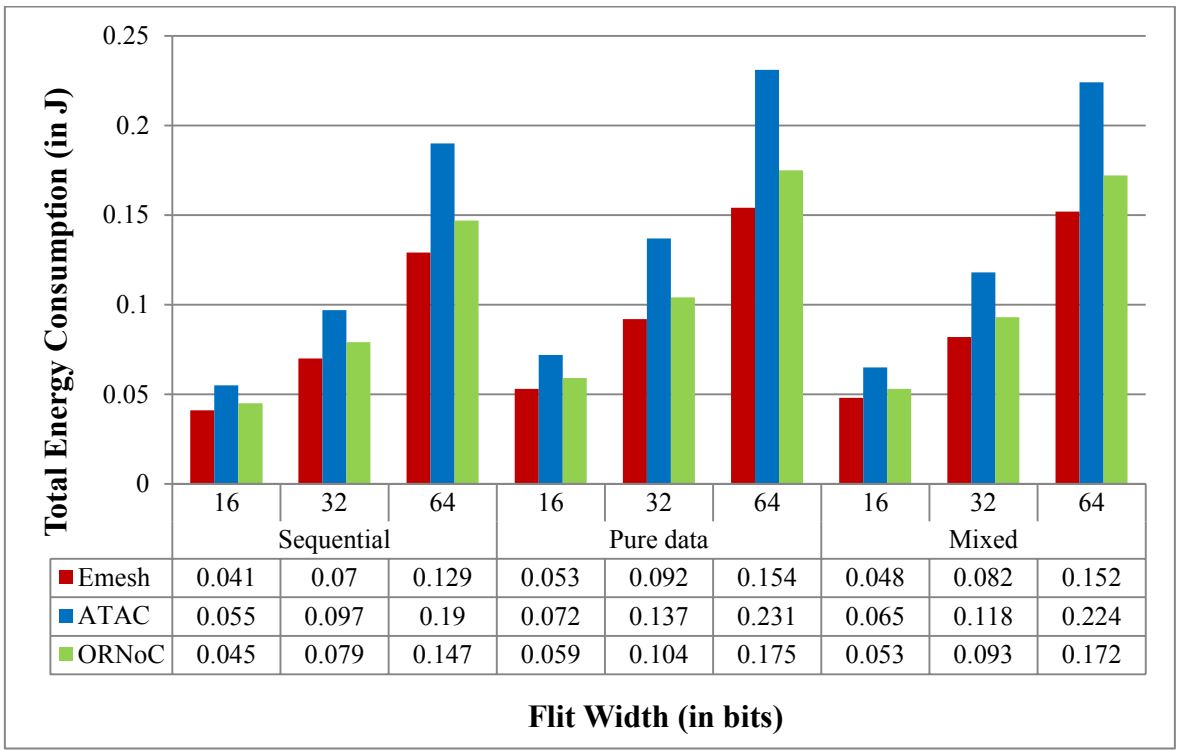


Figure 84: Relative Dynamic Energy variation for Flit Widths N = {16, 32, 64}

The relative variation against Emesh for both the ONoC architectures is similar in all the different implementations for all the different flit widths. The average relative variations are calculated for all the flit width data points considered. The actual and average improvement for all the flit widths in the results is observed to be 50% for ATAC and 55% for ORNoC for all the three implementations against Emesh. Emesh displays higher dynamic energy consumption as compared to both the ONoC architectures as the larger number of electrical links in Emesh constitutes an increase in the dynamic energy consumption.

5.6.5 Results for Total Energy for varying Flit Width Values



	16	32	64	16	32	64	16	32	64
	Sequential			Pure data			Mixed		
Emesh	0.041	0.07	0.129	0.053	0.092	0.154	0.048	0.082	0.152
ATAC	0.055	0.097	0.19	0.072	0.137	0.231	0.065	0.118	0.224
ORNoC	0.045	0.079	0.147	0.059	0.104	0.175	0.053	0.093	0.172

Figure 85: Total Energy Consumption for Flit Widths N = {16, 32, 64}

The total energy consumption values are predominantly constituted by the static energy consumption as the contribution from the dynamic energy consumption is much lesser. The results follow a trend which shows that Emesh architecture has the best total energy, followed by ORNoC and finally by the ATAC architecture. With the lesser wavelengths and waveguides used, the ORNoC architecture outperforms ATAC in the case of total energy consumption. The values increase with increasing flit widths for all the architectures for all the implementations. For the Emesh architecture, the total energy consumption increases from 0.041J when the flit width is 16 bits to 0.129J for 64 bits in the case of sequential implementation, while it is between 0.053J and 0.154J for pure data parallelism implementation for 16 bits and 64 bits respectively and the values are between 0.048J for 16

bits and 0.152J for the 64 bits for the mixed implementation. The ATAC architecture shows the static energy values from 0.055J when the flit width is 16 bits and ranges to 0.19J for 64 bits in the case of sequential implementation, while it is between 0.072J and 0.231J for pure data parallelism implementation for 16 bits and 64 bits respectively and the values are between 0.065J for 64 bits and 0.224J for 64 bits for the mixed implementation. For the ORNoC architecture, the static energy values increase from 0.045J when the flit width is 16 bits to 0.147J for 64 bits in the case of sequential implementation, while it is between 0.059J and ranges to 0.175J for pure data parallelism implementation for 16 bits and 64 bits respectively and the values are between 0.053J for 16 bits and 0.172J for 64 bits in the case of mixed parallelism implementation.

The average values are calculated for all the flit width data points for all the implementations. The average values of total energy consumption are 0.08J for Emesh, 0.114J for ATAC and 0.09J for ORNoC architectures in the case of sequential implementation. The average values for pure data parallelism implementation is 0.1J, 0.147J and 0.113J for Emesh, ATAC and ORNoC architectures respectively, while the mixed implementation gives 0.094J for Emesh and 0.136J for ATAC and 0.106J for ORNoC architectures as the average values for static energy consumption. The pure data implementation shows an average variation of -25% while the mixed implementation shows an average decline of -17.5% with respect to the sequential implementation for the Emesh architecture, while the average degeneration against sequential implementation is -28.95% and -19.30% for pure data parallelism and mixed parallelism implementations respectively for ATAC architecture and the variation is -25.56% for pure data parallelism implementation and -17.78% for the mixed parallelism implementation for the ORNoC architecture.

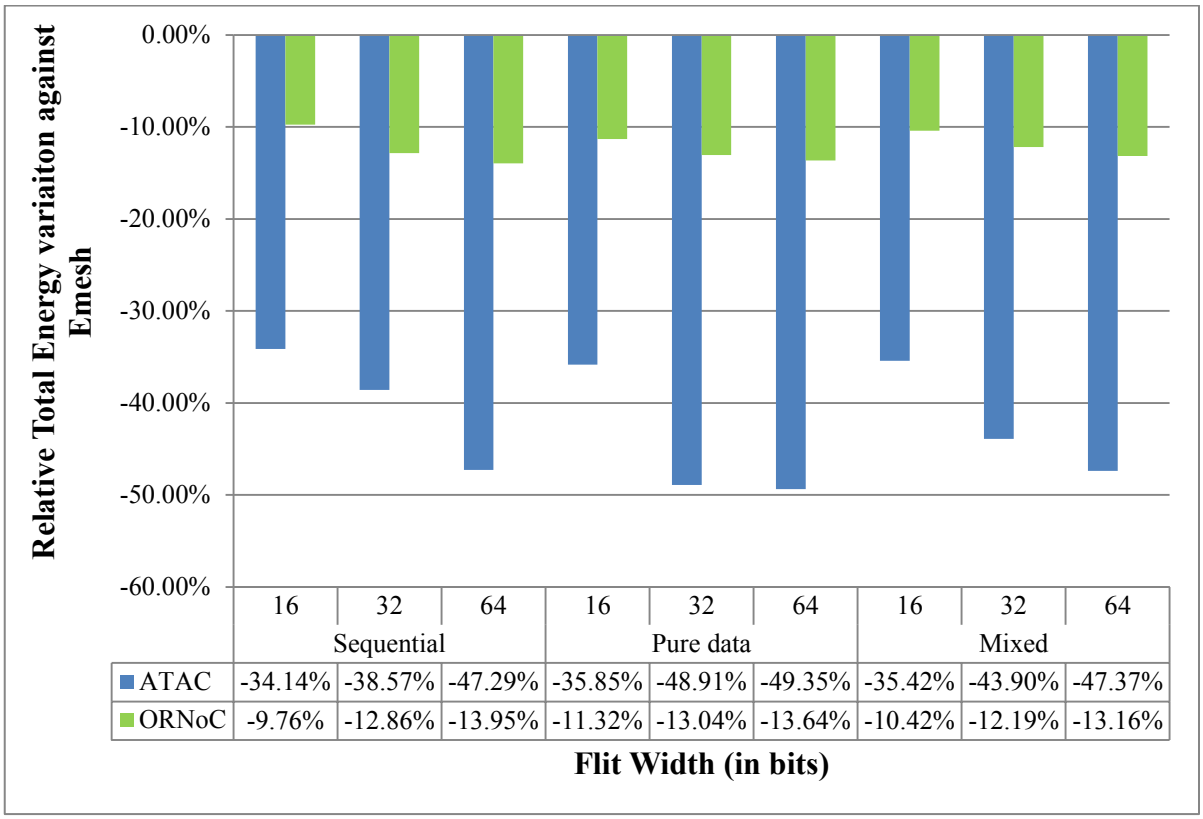


Figure 86: Relative Total Energy variation for Flit Widths N = {16, 32, 64}

The relative total energy variation in the ATAC architecture ranges from -34.14% for 16 bits to -47.29% in the case of 64 bits for sequential implementation, -35.85% for 16 bits to -49.35% in the case of pure data parallelism and -35.42% to -47.37% respectively for 16 bits and 64 bits in the case of mixed parallelism implementation. The ORNoC architecture shows relative variation in total energy from -9.76% for 16 bits to -13.95% in the case of 64 bits for sequential implementation, -11.32% for 16 bits to -13.64% in the case of pure data parallelism and -10.42% to -13.16% respectively for 16 bits and 64 bits in the case of mixed parallelism implementation.

The average relative variations are calculated for all the flit width data points for all the implementations. The ATAC architecture shows average degeneration rates of -40% and ORNoC shows degeneration rates of -12.19% for sequential implementation. The average

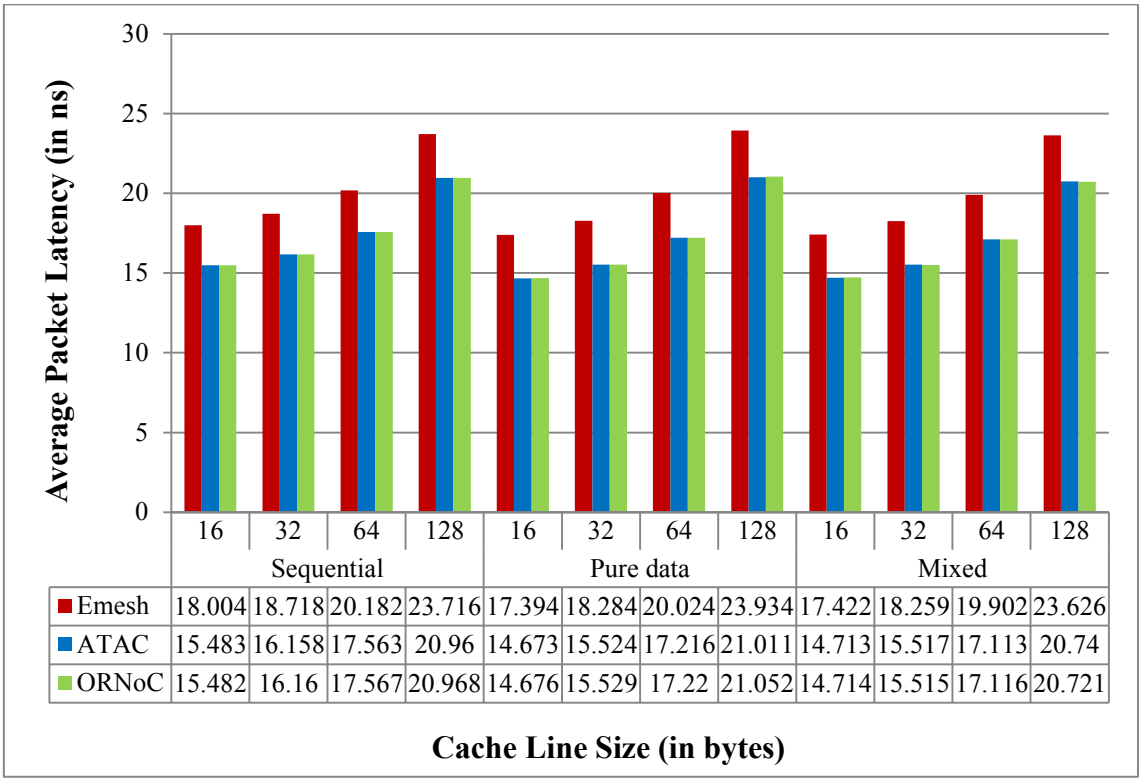
decline rates in the case of pure data parallelism implementation are -44.7% for ATAC and -12.67% for ORNoC architectures. The mixed parallelism implementation shows average decline rates of -42.23% and -11.92% for ATAC and ORNoC architectures respectively. As clear from the previous plots, the ORNoC architecture has lesser degradation in total energy values as compared to the ATAC architecture for all the cases. Also, the degradation tends to increase with increasing flit widths for both the ONoC architectures.

5.7 Analysis with varying Cache Line Size

The cache line size determines the size of the data block transferred from the memory to the cache in a single transfer. It determines how quickly data can be transferred into and out of the cache. The required data for each core is transferred from the shared memory to their corresponding caches. With the implementation of parallelization schemes, smaller data blocks are required to be transferred per core, eliminating the overhead for transfer of the whole image in sequential implementation. Since this thesis uses only shared memory communication and not message passing, the data is transferred from the shared memory to the respective caches of each core. The experiment analyzes how the different cache line sizes perform for the chosen workload. The cache line size in Graphite simulator is set up in such a way that, it is the same for all the caches for the architectures, which are the L1 data cache, L1 instruction cache and a shared L2 cache mainly for storing data. One extra parameter observed for this set of experiments as well as the one with varying cache size is the miss rate for L1 data cache. The L1 data cache is considered because it is the primary data cache and has the primal influence over the faster processing.

The cache line size values considered are 16, 32, 64 and 128 bytes. The set of experiments use a test image of 512 X 512 pixels (237 kB) and is executed on 64 cores to observe the results. The smallest image size is used as the cache line size are in bytes and hence, the variations will be noticeable for this image size itself without much overhead in simulation time. Also, observations with larger image sizes have not yielded significant improvement in outputs for this set of experiments. After observing the results for various cluster numbers, the number of clusters chosen is 4 as the increased number of clusters don't contribute any significant improvement in the delay and energy properties for this set of experiments. It also highlights that with minimum optical communication in the network there is improvement in the delay and energy parameters for variation in the cache line size. The specific configurations used for this experiment is mentioned in Appendix Table 7.

5.7.1 Results for Packet Latency for varying Cache Line Sizes



	16	32	64	128	16	32	64	128	16	32	64	128
	Sequential				Pure data				Mixed			
Emesh	18.004	18.718	20.182	23.716	17.394	18.284	20.024	23.934	17.422	18.259	19.902	23.626
ATAC	15.483	16.158	17.563	20.96	14.673	15.524	17.216	21.011	14.713	15.517	17.113	20.74
ORNoC	15.482	16.16	17.567	20.968	14.676	15.529	17.22	21.052	14.714	15.515	17.116	20.721

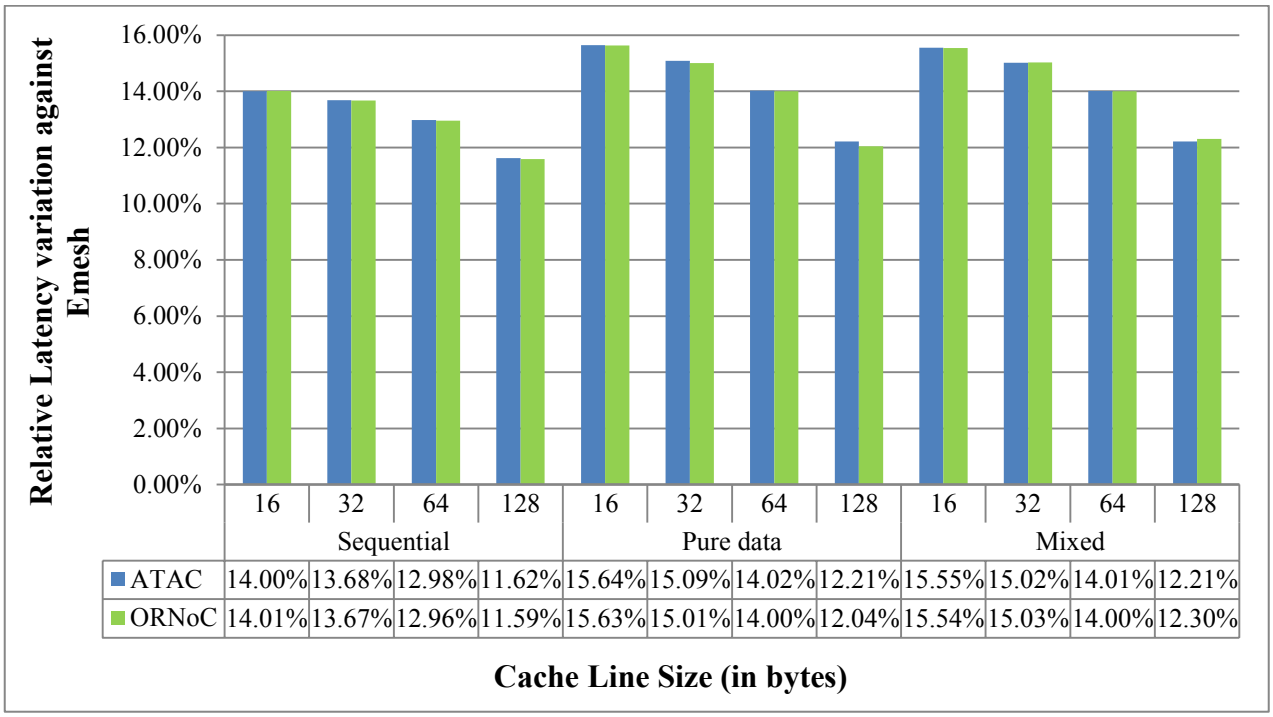
Figure 87: Average Packet Latency for Cache Line Sizes N = {16, 32, 64, 128}

The average packet latency values are observed to increase with increase in the cache line size. The increased cache line size may cause transfer of larger amount of data which are not required for the current operation. For instance, initially when a convolution is being performed on the first block, it requires only image data pertaining to the first block and not the others which might be fetched into the cache and hence, leads to increased time in fetching data that will not be used. For the Emesh architecture, the packet latency increases from 18.004ns when the cache line size is 16 bytes to 23.716ns for a cache line size of 128 bytes in the case of sequential implementation, while it is between 17.394ns and varies to 23.934ns for pure data parallelism implementation for 16 bytes and 128 bytes respectively and the values are between 17.422ns for 16 bytes and 23.626ns for 128 bytes in the case of

mixed implementation. The ATAC architecture shows the packet latency values from 15.483ns when the cache line size is 16 bytes to 20.96ns for 128 bytes in the case of sequential implementation, while it is between 14.673ns and range to 21.011ns for pure data parallelism implementation for 16 byte cache line size and 128 byte cache line size respectively and the values are between 14.713ns for 16 byte cache line size and 20.74ns for the 128 byte cache line size for the mixed implementation. For the ORNoC architecture, the packet latency values increase from 15.482ns when the cache line size is 16 bytes to 20.968ns for 128 bytes in the case of sequential implementation, while it is between 14.676ns and varies to 21.052ns for pure data parallelism implementation for 16 byte cache line size and 128 byte cache line size respectively and the values are between 14.714ns for 16 byte cache line size and 20.721ns for the 128 byte cache line size for the mixed implementation.

The sequential implementation shows average latency values of 20.16ns for Emesh and 17.54ns for ATAC and ORNoC architectures respectively for all the cache line sizes. The average values for Emesh are 19.91ns, 17.11ns for ATAC and 17.12ns for ORNoC architectures in the case of pure data parallelism implementation for all the cache line sizes considered. The mixed implementation scheme gives the average values of 19.80ns for Emesh, 17.02ns for both ATAC and ORNoC architectures for all the cache line sizes. The latency values for Emesh are observed to be greater than that of the ONoC architectures. The average variations of the pure data parallelism implementation and the mixed parallelism implementations against sequential implementation are: 1.24% and 1.79% respectively for Emesh, 2.45% and 2.96% respectively for ATAC, 2.39% and 2.96% respectively for ORNoC architectures for all the cache line sizes.

The parallelization schemes have higher values for latency in the order of mixed parallelization scheme with the least values for latency followed by pure data parallelization and finally by the sequential implementation which has the highest value except when the cache line size is 128 bytes. The efficiency in implementing parallelization reduces the latency values for the parallelized implementations and hence, sequential implementation has greater packet latency. Both the parallelized implementations have latency values very close to each other.



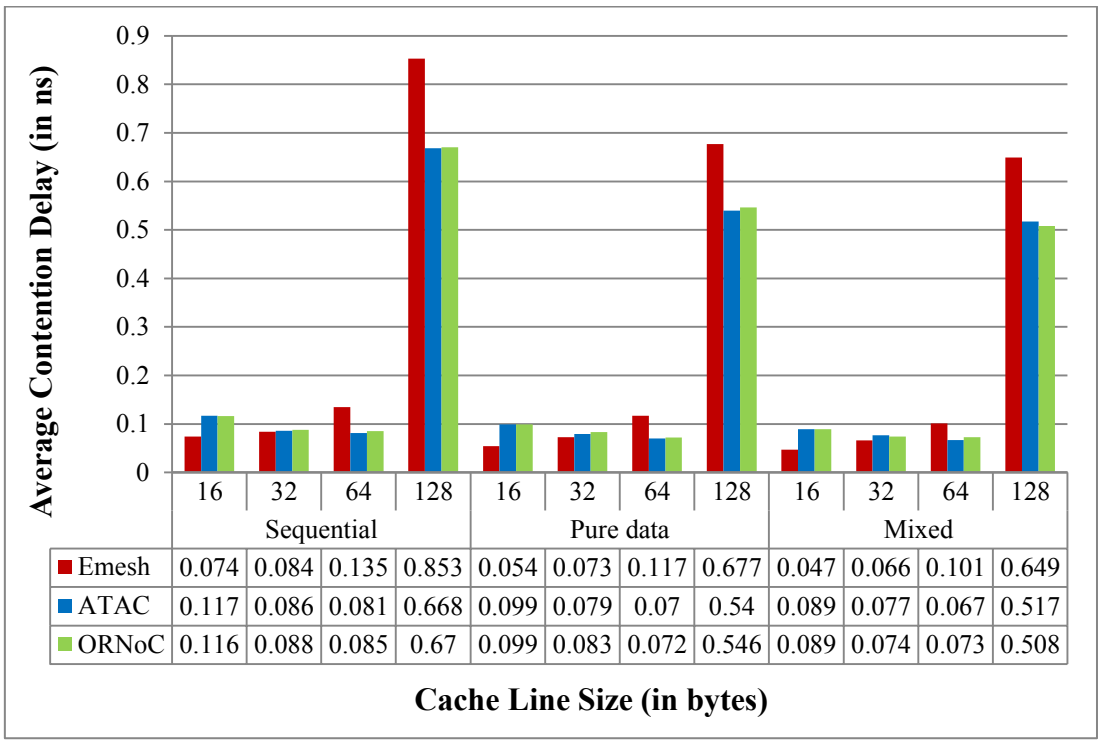
	16	32	64	128	16	32	64	128	16	32	64	128
	Sequential				Pure data				Mixed			
ATAC	14.00%	13.68%	12.98%	11.62%	15.64%	15.09%	14.02%	12.21%	15.55%	15.02%	14.01%	12.21%
ORNoC	14.01%	13.67%	12.96%	11.59%	15.63%	15.01%	14.00%	12.04%	15.54%	15.03%	14.00%	12.30%

Figure 88: Relative latency variation for Cache Line Sizes N = {16, 32, 64, 128}

As average packet latency is observed to increase with increase in cache line size, the relative latency variation with respect to Emesh decreases. The relative packet latency values for ATAC are 14% for a cache line size of 16 bytes and goes to 11.62% for a cache line size of 128 bytes in the case of sequential implementation, 15.64% for 16 bytes and goes to 12.21% in the case of pure data parallelism implementation and 15.55% for cache line size of 16 bytes and goes to 12.21% in the case of 128 bytes in the case of mixed parallelism

implementation. The ORNoC architecture shows relative contention delay values of 14.01% for a cache line size of 16 bytes and goes to 11.59% for a cache line size of 128 bytes in the case of sequential implementation, 15.63% for 16 bytes and goes to 12.04% in the case of pure data parallelism implementation and 15.54% for cache line size of 16 bytes and goes to 12.30% in the case of 128 bytes in the case of mixed parallelism implementation. The average improvement in latency rates against Emesh is 13.07% for ATAC and 13.06% for ORNoC in the case of sequential implementation, 14.24% and 14.17% for ATAC and ORNoC architectures respectively for pure data parallelism implementation, 14.20% for ATAC and 14.22% for ORNoC architectures in the case of the mixed parallelism implementation. Emesh architecture shows much higher latency as it only has electrical networks for communication, while both the ONoC architectures have better latency values than Emesh.

5.7.2 Results for Contention Delay for varying Cache Line Sizes



	16	32	64	128	16	32	64	128	16	32	64	128
	Sequential				Pure data				Mixed			
Emesh	0.074	0.084	0.135	0.853	0.054	0.073	0.117	0.677	0.047	0.066	0.101	0.649
ATAC	0.117	0.086	0.081	0.668	0.099	0.079	0.07	0.54	0.089	0.077	0.067	0.517
ORNoC	0.116	0.088	0.085	0.67	0.099	0.083	0.072	0.546	0.089	0.074	0.073	0.508

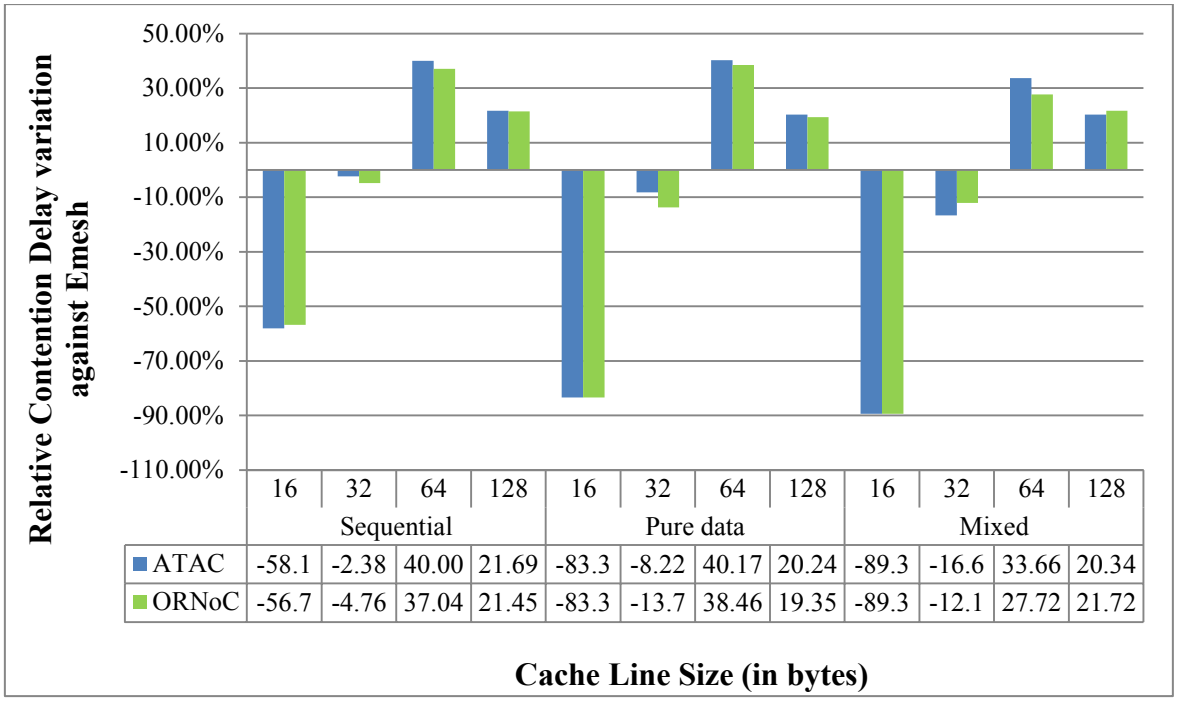
Figure 89: Average Contention Delay for Cache Line Sizes N = {16, 32, 64, 128}

The contention delay values are observed to be smaller for Emesh for smaller cache line size and increases above that of the ONoC architectures with the increase in cache line size. The ONoC architectures show greater values initially and transit to smaller contention delay values for 32 and 64 bytes before increasing when the cache line size is 128 bytes. The initial fetch of data which is not present in the cache memory, can lead to fetching of data which is too long due to the bigger cache line and leads to cache pollution, while studies also show that the contention in the network for long cache sizes increases proportional to the square of message length[55]. The contention delay for ONoC architectures improve with increasing cache line size than the Emesh architecture. For the Emesh architecture, the contention delay increases from 0.074ns when the cache line size is 16 bytes to 0.853ns for a

cache line size of 128 bytes in the case of sequential implementation, while it is between 0.054ns and varies to 0.677ns for pure data parallelism implementation for 16 bytes and 128 bytes respectively and the values are between 0.047ns for 16 bytes and 0.649ns for 128 bytes in the case of mixed implementation. The ATAC architecture shows the contention delay values from 0.117ns when the cache line size is 16 bytes to 0.668ns for 128 bytes in the case of sequential implementation, while it is between 0.099ns and range to 0.54ns for pure data parallelism implementation for 16 byte cache line size and 128 byte cache line size respectively and the values are between 0.089ns for 16 byte cache line size and 0.517ns for the 128 byte cache line size for the mixed implementation. For the ORNoC architecture, the contention delay values increase from 0.116ns when the cache line size is 16 bytes to 0.67ns for 128 bytes in the case of sequential implementation, while it is between 0.099ns and varies to 0.546ns for pure data parallelism implementation for 16 byte cache line size and 128 byte cache line size respectively and the values are between 0.089ns for 16 byte cache line size and 0.508ns for the 128 byte cache line size for the mixed implementation.

The average values of contention delay for sequential implementation are 0.243ns, 0.117ns and 0.118ns for Emesh, ATAC and ORNoC architectures respectively for all the cache line sizes considered. Pure data parallelism implementation gives average values of 0.201ns for Emesh, 0.096ns for ATAC and 0.095ns for ORNoC architectures for all the cache line sizes considered. The average values for mixed implementation are 0.188ns, 0.098ns and 0.093ns respectively for Emesh, ATAC and ORNoC architectures for all the cache line sizes considered. The average variation between the pure data parallelism and mixed parallelism against sequential implementation are 17.82% and 22.63% for Emesh, 17.95% and 16.24% for ATAC and 19.49% and 21.19% for ORNoC respectively for all the cache line sizes.

The parallelization schemes have the values of contention delay in the order of mixed parallelization scheme with the least contention delay followed by pure data parallelization and finally by the sequential implementation which has the highest value. Similar to packet latency, the efficient implementation of the parallelization schemes improves the contention delay values of the parallelized implementations over sequential implementation. Both the parallelized implementations have contention delay values very close to each other.



	Sequential				Pure data				Mixed			
	16	32	64	128	16	32	64	128	16	32	64	128
ATAC	-58.1	-2.38	40.00	21.69	-83.3	-8.22	40.17	20.24	-89.3	-16.6	33.66	20.34
ORNoC	-56.7	-4.76	37.04	21.45	-83.3	-13.7	38.46	19.35	-89.3	-12.1	27.72	21.72

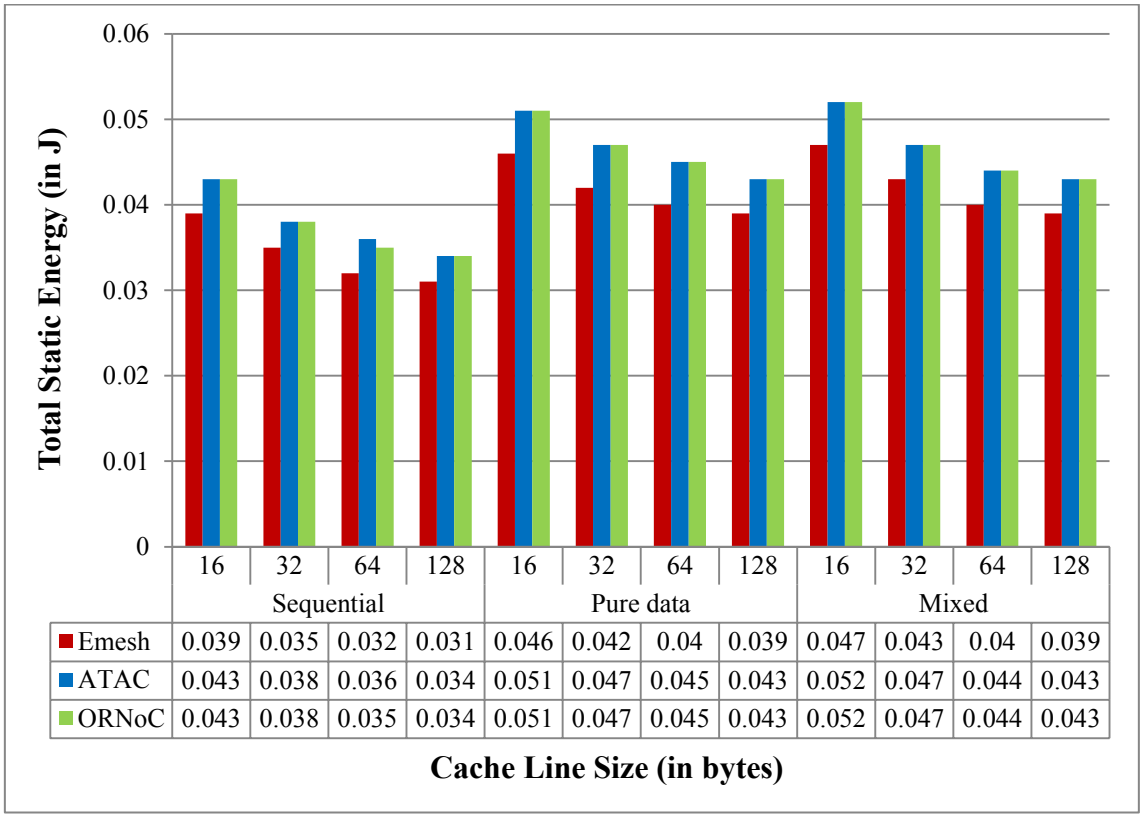
Figure 90: Relative Contention Delay variation for Cache Line Sizes N = {16, 32, 64, 128}

The relative variation in cache size shows that the values improve for both ONoC architectures until the cache line size is 64 bytes and decreases again after that. Emesh architecture shows much higher delays as it only has electrical networks for communication, while both the ONoC architectures have better contention delay values than Emesh. The relative contention delay values for ATAC are -58.1% for a cache line size of 16 bytes and improves to 21.69% for a cache line size of 128 bytes in the case of sequential

implementation, -83.3% for 16 bytes and goes to 20.24% in the case of pure data parallelism implementation and -89.33% for cache line size of 16 bytes and improve to 20.34% in the case of 128 bytes in the case of mixed parallelism implementation. The ORNoC architecture shows relative contention delay values of -56.7% for a cache line size of 16 bytes and improves to 21.45% for a cache line size of 128 bytes in the case of sequential implementation, -83.3% for 16 bytes and goes to 19.35% in the case of pure data parallelism implementation and -89.33% for cache line size of 16 bytes and improve to 21.72% in the case of 128 bytes in the case of mixed parallelism implementation.

The average improvement in latency rates against Emesh is 30.25% for ATAC and -0.74% for ORNoC in the case of sequential implementation, -7.75% and -9.81% for ATAC and ORNoC architectures respectively for pure data parallelism implementation, -12.99% for ATAC and -12.99% for ORNoC architectures in the case of the mixed parallelism implementation. Emesh architecture shows much higher latency as it only has electrical networks for communication, while both the ONoC architectures have better latency values than Emesh. The contention delay values for both the ONoC architectures show values close to each other.

5.7.3 Results for Static Energy for varying Cache Line Sizes



	16	32	64	128	16	32	64	128	16	32	64	128
	Sequential				Pure data				Mixed			
Emesh	0.039	0.035	0.032	0.031	0.046	0.042	0.04	0.039	0.047	0.043	0.04	0.039
ATAC	0.043	0.038	0.036	0.034	0.051	0.047	0.045	0.043	0.052	0.047	0.044	0.043
ORNoC	0.043	0.038	0.035	0.034	0.051	0.047	0.045	0.043	0.052	0.047	0.044	0.043

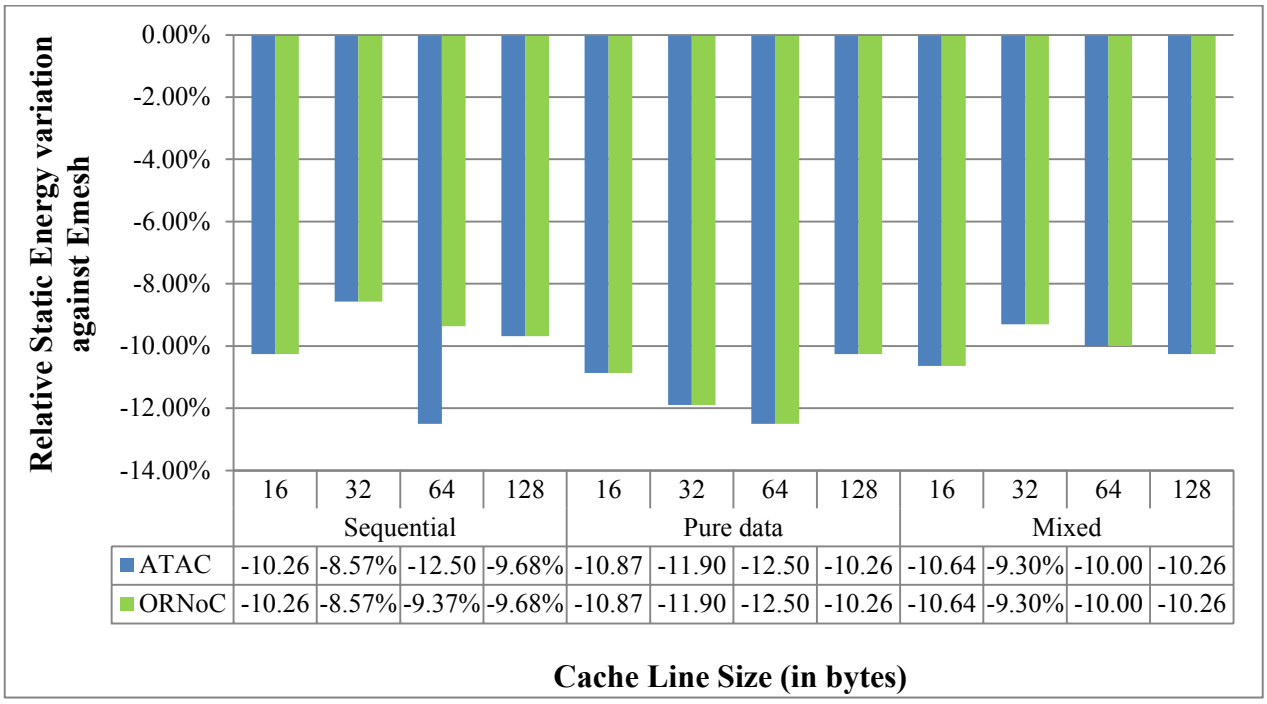
Figure 91: Total Static Energy for Cache Line Sizes N = {16, 32, 64, 128}

The static energy decreases with increasing cache line size for all network architectures. The mixed parallelization scheme has the highest set of values among the three implementations while the sequential implementation has the lowest. The Emesh architecture is observed to have lesser static energy consumption as compared to the ONoC architectures as the ONoC architecture have the presence of optical components that consume more non-data dependent energy. For the Emesh architecture, the static energy decreases from 0.039J when the cache line size is 16 bytes to 0.031J for a cache line size of 128 bytes in the case of sequential implementation, while it is between 0.046J and varies to 0.039J for pure data parallelism implementation for 16 bytes and 128 bytes respectively and the values are between 0.047J

for 16 bytes and 0.039J for 128 bytes in the case of mixed implementation. Both the ATAC and ORNoC architectures show the static energy values from 0.043J when the cache line size is 16 bytes to 0.034J for 128 bytes in the case of sequential implementation, while it is between 0.051J and range to 0.043J for pure data parallelism implementation for 16 byte cache line size and 128 byte cache line size respectively and the values are between 0.052J for 16 byte cache line size and 0.043J for the 128 byte cache line size for the mixed implementation.

The average values of static energy consumption for all the cache line sizes for sequential implementation are 0.034J, 0.038J and 0.037J for Emesh, ATAC and ORNoC architectures respectively. Pure data parallelism implementation and mixed parallelism implementation gives average values of 0.042J for Emesh and 0.046J for ATAC and ORNoC architectures for all the cache line sizes considered. The average variation between the pure data parallelism and mixed parallelism against sequential implementation are -23.53% for Emesh, -21.05% for ATAC and -24.32% for ORNoC for all the cache line sizes.

The static energy consumption is greater for the parallelized schemes as compared to the sequential implementation. The number of data packets in the network in the case of sequential implementation is lesser than those for both the parallelized implementations for varying cache line sizes. This increases the usage of components that contribute towards static energy consumption and hence, the parallelized implementations have higher static energy consumption.



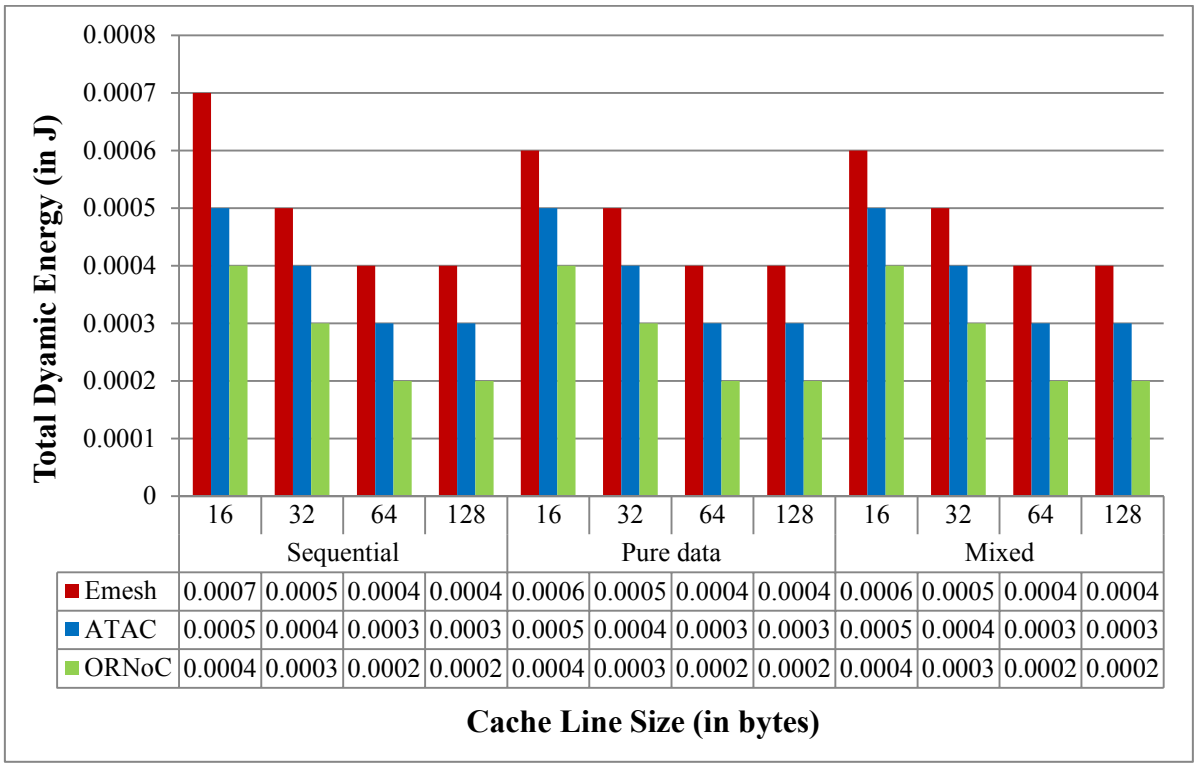
	16	32	64	128	16	32	64	128	16	32	64	128
	Sequential				Pure data				Mixed			
ATAC	-10.26	-8.57%	-12.50	-9.68%	-10.87	-11.90	-12.50	-10.26	-10.64	-9.30%	-10.00	-10.26
ORNoC	-10.26	-8.57%	-9.37%	-9.68%	-10.87	-11.90	-12.50	-10.26	-10.64	-9.30%	-10.00	-10.26

Figure 92: Relative Static Energy variation for Cache Line Sizes N = {16, 32, 64, 128}

The figure shows the relative variation in static energy of the ONoC architectures with respect to the Emesh architecture. The relative variations, change by around 3% for a change in the data point by 0.001 for this experiment. For both ATAC and ORNoC architectures, the relative variation in the static energy values is from -10.2% for cache line size of 16 bytes to -9.68% for cache line size of 128 bytes in the case of sequential implementation, from -10.8% for cache line size of 16 bytes to -10.2% for cache line size of 128 bytes in the case of pure data parallelism implementation and from -10.6% for cache line size of 16 bytes to -10.2% for cache line size of 128 bytes. The average decrease for all the cache line sizes in the static energy consumption for sequential implementation is -10.24% for ATAC and -9.46% for ORNoC. For the pure data parallelism implementation and mixed parallelism implementation, it is -11.35% and -10.02% for both the ONoC architectures respectively for all the cache line sizes. Optical network architectures are outperformed by Emesh in static

energy consumption as it is lesser without any optical components in the network [54]. Both the ONoC architectures show similar variations in most of the cases, with the exception of one case where ORNoC has better values than ATAC. Among the optical networks, ORNoC architecture has lesser static energy as the number of photonic components that require more electrical circuitry support is lesser than in ATAC due to wavelength reuse [54].

5.7.4 Results for Dynamic Energy for varying Cache Line Sizes



	Sequential				Pure data				Mixed			
	16	32	64	128	16	32	64	128	16	32	64	128
Emesh	0.0007	0.0005	0.0004	0.0004	0.0006	0.0005	0.0004	0.0004	0.0006	0.0005	0.0004	0.0004
ATAC	0.0005	0.0004	0.0003	0.0003	0.0005	0.0004	0.0003	0.0003	0.0005	0.0004	0.0003	0.0003
ORNoC	0.0004	0.0003	0.0002	0.0002	0.0004	0.0003	0.0002	0.0002	0.0004	0.0003	0.0002	0.0002

Figure 93: Total Dynamic Energy for Cache Line Sizes N = {16, 32, 64, 128}

The total dynamic energy consumption is also observed to decrease with increase in cache line size. The algorithm uses more spatial localization as compared to temporal localization. Hence, the data that lie closer to each other in space is expected to be reused more frequently and favour bigger cache line size. This in turn increases the hit rates and reduces the data dependent energy consumption. For the Emesh architecture, the dynamic energy decreases

from 0.0007J when the cache line size is 16 bytes to 0.0004J for a cache line size of 128 bytes in the case of sequential implementation, while it is between 0.0006J and varies to 0.004J for both pure data parallelism implementation and mixed parallelism for 16 bytes and 128 bytes respectively. The ATAC architecture shows dynamic energy values from 0.0005J when the cache line size is 16 bytes to 0.0003J for 128 bytes in the case of all the three implementations, while it is between 0.0004J and ranges to 0.0002J for 16 byte cache line size and 128 byte cache line size respectively in the case of ORNoC architecture for all the three implementations.

The average dynamic energy consumption values for all the cache line sizes in the case of Emesh are 0.0005J in the case of sequential implementation and 0.0004J for both pure data and mixed parallelism implementations. The average values for all the cache line sizes are 0.0004J for ATAC and 0.0003J for ORNoC for all the three different implementation schemes. The different implementation schemes have similar dynamic energy consumption as they work on the same amount of data, which constitutes the dynamic energy consumption.

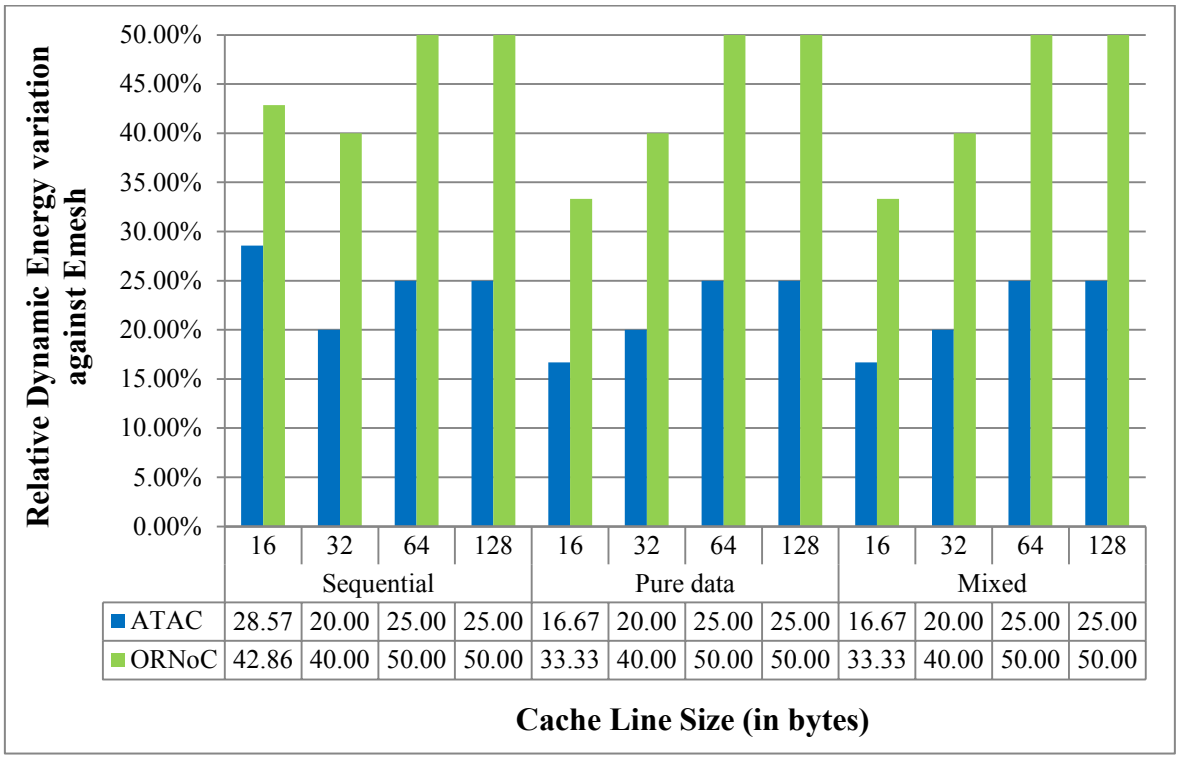


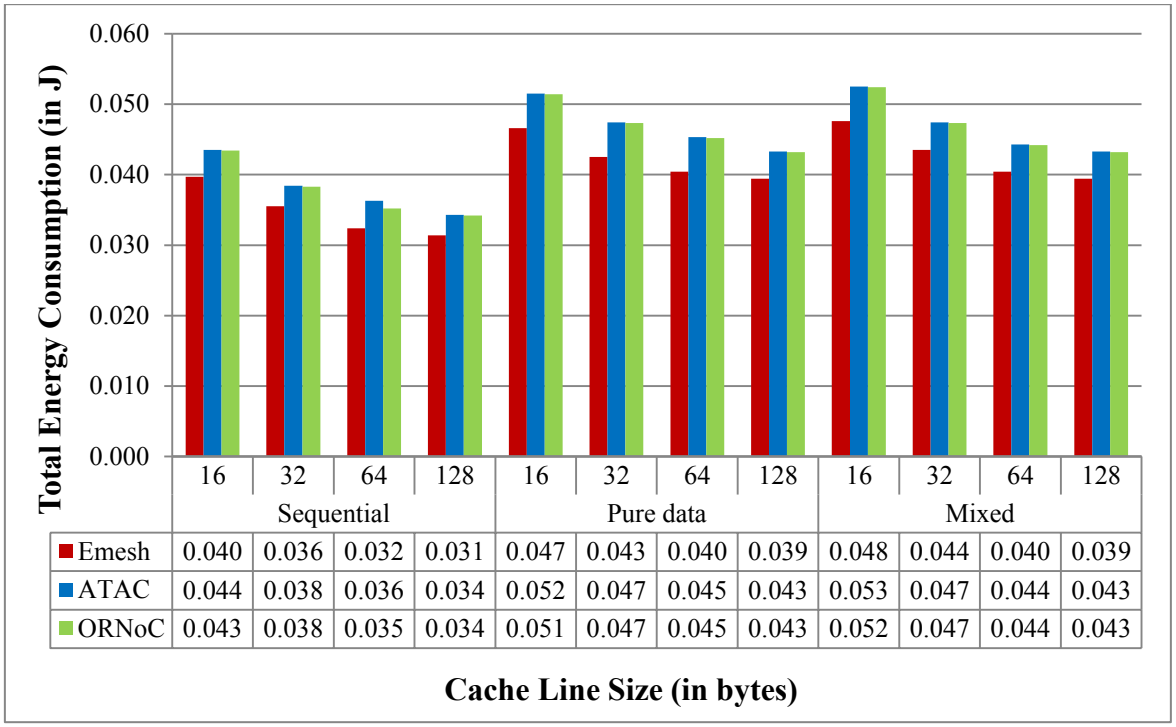
Figure 94: Relative Dynamic Energy variation for Cache Line Sizes N = {16, 32, 64, 128}

The variation in dynamic energy against Emesh increases for both the ONoC architectures for the pure data and mixed parallelization schemes. The relative dynamic energy values for ATAC architecture are: 28.57% for the cache line size of 16 bytes and goes to 25% for cache line size of 128 bytes in the case of sequential implementation, 16.67% for 16 bytes and goes to 25% for 128 bytes in the case of both pure data parallelism implementation and mixed parallelism implementation. The ORNoC architecture shows relative contention delay values of 42.86% for a cache line size of 16 bytes and improves to 50% for a cache line size of 128 bytes in the case of sequential implementation, 33.33% for 16 bytes and goes to 50% for 128 bytes in the case of both pure data parallelism implementation and mixed parallelism implementation.

The average variation in dynamic energy consumption against Emesh for all the cache line sizes in the case of sequential implementation is 24.64% and 45.72% for ATAC and ORNoC

architectures respectively. Whereas, both the pure data parallelism and mixed implementation schemes show 21.67% average improvement for ATAC and 43.33% average improvement for ORNoC for both pure data parallelism and mixed parallelism implementations while considering all the cache line sizes. Emesh architecture has more number of electrical links that contribute to the dynamic energy and hence, it is greater for Emesh architecture as compared to ONoC architectures [54]. ORNoC architecture uses lesser number of wavelengths and waveguides which in turn lead to decreased number of modulators and receivers that contribute to data dependent energy consumption and hence, has slightly better dynamic energy consumption as compared to ATAC [54].

5.7.5 Results for Total Energy for varying Cache Line Sizes



	16	32	64	128	16	32	64	128	16	32	64	128
	Sequential				Pure data				Mixed			
Emesh	0.040	0.036	0.032	0.031	0.047	0.043	0.040	0.039	0.048	0.044	0.040	0.039
ATAC	0.044	0.038	0.036	0.034	0.052	0.047	0.045	0.043	0.053	0.047	0.044	0.043
ORNoC	0.043	0.038	0.035	0.034	0.051	0.047	0.045	0.043	0.052	0.047	0.044	0.043

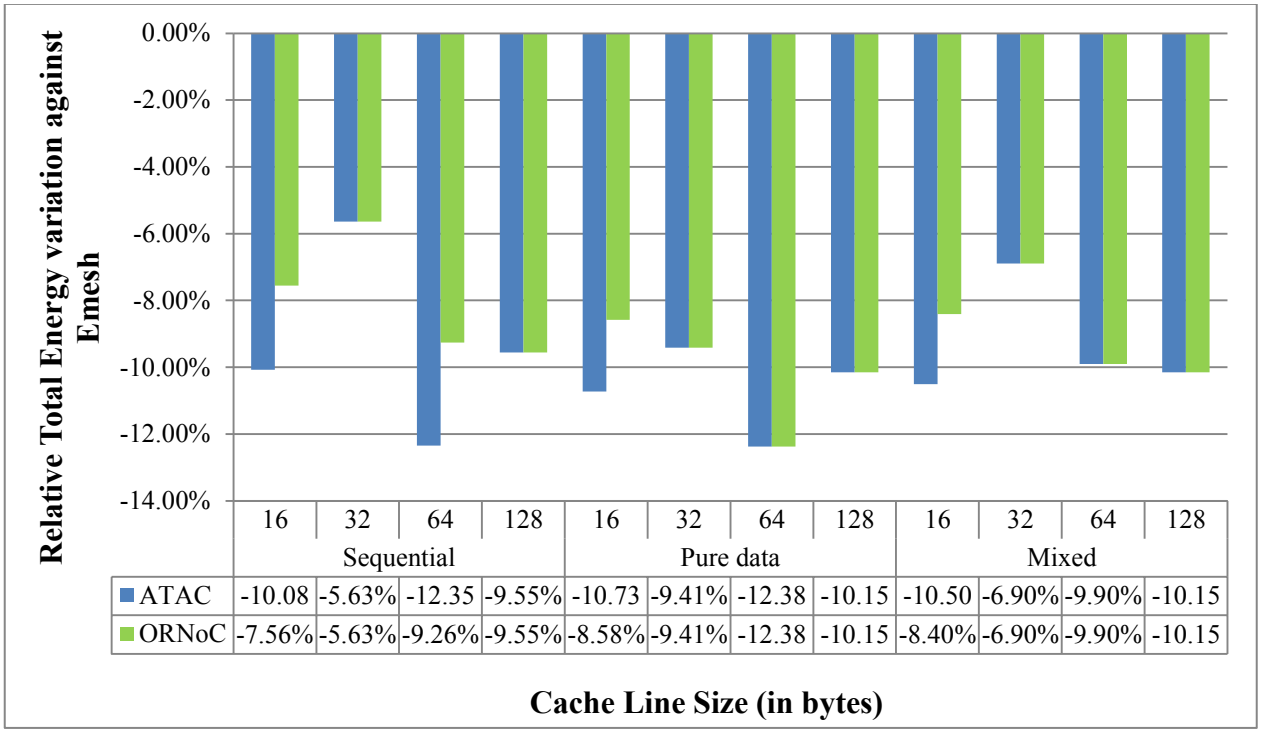
Figure 95: Total Energy Consumption for Cache Line Sizes N = {16, 32, 64, 128}

The total energy consumption follows the same decreasing trend with increasing cache line sizes. The dynamic energy consumption values contribute a very minor fraction of the total

energy consumption, while majority of it is contributed by the static energy consumption. For the Emesh architecture, the total energy consumption increases from 0.04J when the cache line size is 16 bytes to 0.031J for cache line size of 128 bytes in the case of sequential implementation, while it is between 0.047J and varies to 0.039J for pure data parallelism implementation for 16 bytes and 128 bytes respectively and the values are between 0.048J for 16 bytes and 0.039J for 128 bytes in the case of mixed implementation. The ATAC architecture shows the total energy consumption values from 0.044J when the cache line size is 16 bytes to 0.034J for 128 bytes in the case of sequential implementation, while it is between 0.052J and range to 0.43J for pure data parallelism implementation for 16 byte cache line size and 128 byte cache line size respectively and the values are between 0.053J for 16 byte cache line size and 0.043J for the 128 byte cache line size for the mixed implementation. For the ORNoC architecture, the total energy consumption values increase from 0.043J when the cache line size is 16 bytes to 0.034J for 128 bytes in the case of sequential implementation, while it is between 0.051J and varies to 0.043J for pure data parallelism implementation for 16 byte cache line size and 128 byte cache line size respectively and the values are between 0.052J for 16 byte cache line size and 0.043J for the 128 byte cache line size for the mixed implementation.

The average values of total energy consumption for sequential implementation are 0.035J, 0.038J and 0.037J for Emesh, ATAC and ORNoC architectures respectively for all the cache line sizes considered. Pure data parallelism implementation and mixed implementation gives average values of 0.042J and 0.043J for Emesh, while both implementations give average value of 0.046J for ATAC and ORNoC architectures for all the cache line sizes. The average variation for all the cache line sizes between the pure data parallelism and mixed parallelism

against sequential implementation are -20% and -22.86% for Emesh, while it is -21.05% for ATAC and -24.32% for ORNoC. Since static energy is the major constituent, the Emesh architecture has better values for total energy consumption as compared to both ONoC architectures. The plots follow the same trend as that of static energy consumption.



	16	32	64	128	16	32	64	128	16	32	64	128
	Sequential				Pure data				Mixed			
ATAC	-10.08	-5.63%	-12.35	-9.55%	-10.73	-9.41%	-12.38	-10.15	-10.50	-6.90%	-9.90%	-10.15
ORNoC	-7.56%	-5.63%	-9.26%	-9.55%	-8.58%	-9.41%	-12.38	-10.15	-8.40%	-6.90%	-9.90%	-10.15

Figure 96: Relative Total Energy variation for Cache Line Sizes N = {16, 32, 64, 128}

Since the major contribution towards total energy is provided by static energy, the graph follows the same pattern as that of static energy consumption for the same reasons. The relative variation in total energy consumption values for ATAC are from -10.08% to -9.55% for 16 byte cache line size and 128 byte cache line size respectively for sequential implementation, -10.73% to -10.15% for 16 byte cache line size and 128 byte cache line size respectively in the case of pure data parallelism implementation and -10.50% for 16 byte cache line size to -10.15% for 128 byte cache line size in the case of mixed parallelism implementation. The relative variation in total energy consumption values for ORNoC are

from -7.56% to -9.55% for 16 byte cache line size and 128 byte cache line size respectively for sequential implementation, -8.58% to -10.15% for 16 byte cache line size and 128 byte cache line size respectively in the case of pure data parallelism implementation and -8.40% for 16 byte cache line size to -10.15% for 128 byte cache line size in the case of mixed parallelism implementation.

The relative decline in total energy fir all the cache line sizes against Emesh shows average values of -9.40% for ATAC and -8% for ORNoC in the case of sequential implementation. The pure data parallelism implementation shows average rates of -10.67% for ATAC and -10.13% for ORNoC for all the cache line sizes. The mixed parallelism implementation average values are -9.36% and -8.84% for ATAC and ORNoC respectively for all the cache line sizes. The total energy decline rate is minimal for all the architectures for the cache line size of 32 bytes with the next smallest variation at 128 bytes. The variations are observed to be the largest in the case of pure data parallelism implementation, followed by the mixed implementation and then by the sequential implementation. The ORNoC variations are observed to be lesser or equal to that of the ATAC architecture for all the implementations.

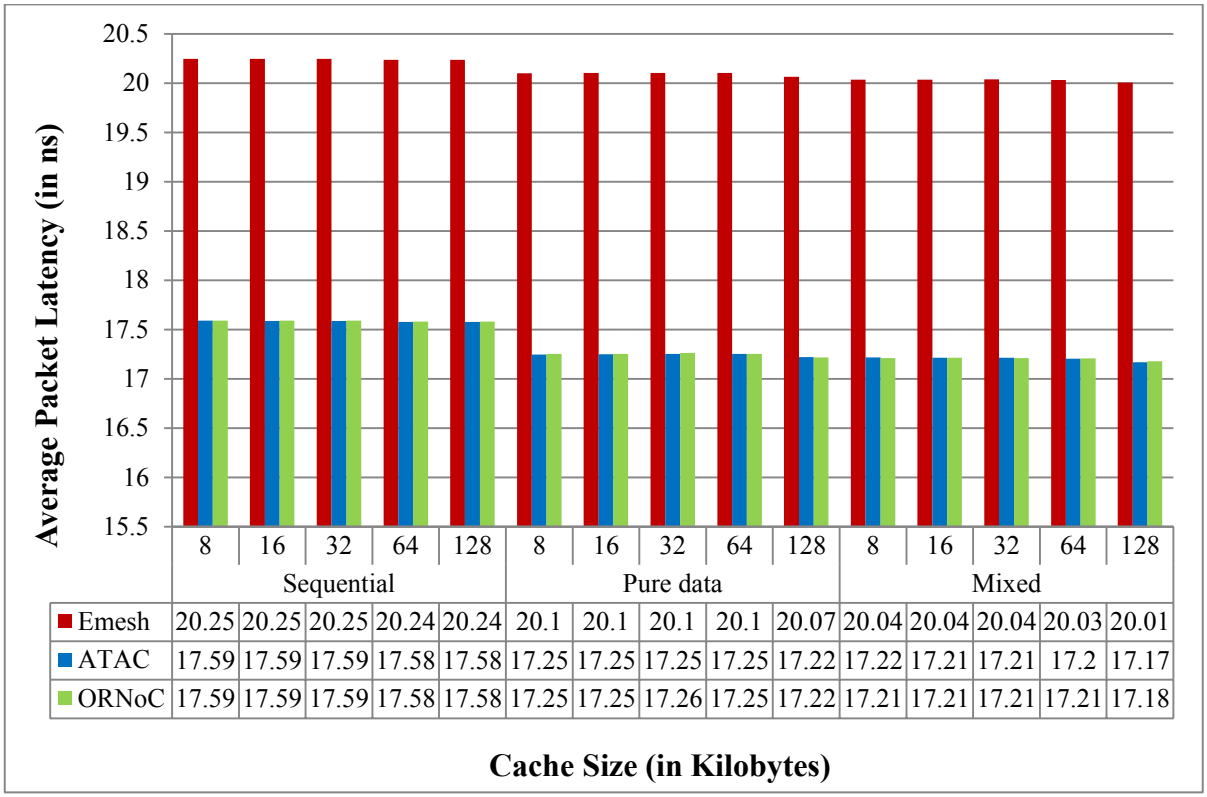
5.8 Analysis with varying Size of L1-Data Cache

L1 or level 1 cache size determines the amount of data that can be at each processing core's immediate disposal, with the use of least number of instruction cycles for the data fetch operation while processing. The cache size is usually determined by the trade-off between extra on-chip area and cost against the hit rates in the case of actual hardware design. This set of experiments evaluates the influence of cache size on network delays and the power consumption of the network. From the observations, it is expected to obtain the optimum L1

cache size in terms of delay and power consumption for the different implementations of the Canny edge detection algorithm.

The values considered for this set of experiments are powers of 2 from 8KB to 128KB, which is a reasonably high value for L1 data cache. The set of experiments use a test image of size 1024 x 1024 pixels and is executed on 64 cores to observe the results. This image size is moderately large as compared to the maximum cache size considered for this set of experiments. This would help in analyzing if the chosen cache size can handle the operation efficiently for the chosen core size. To explain further, for a very large image size, the larger cache size has an advantage or the very small image size in which the smaller cache can handle the operation efficiently. The cluster size considered is 16 cores per cluster. Different cluster sizes (4,8,16) were used to perform the simulations, but most of them gave very similar values for the outputs. The one with the most variation among them was for 4 clusters. Hence, it was chosen for tabulating the results. The rest of the configurations are kept as default and mentioned in Appendix Table 8.

5.8.1 Results for Packet Latency for varying Size of L1-Data Cache



	Sequential					Pure data					Mixed				
	8	16	32	64	128	8	16	32	64	128	8	16	32	64	128
Emesh	20.25	20.25	20.25	20.24	20.24	20.1	20.1	20.1	20.1	20.07	20.04	20.04	20.04	20.03	20.01
ATAC	17.59	17.59	17.59	17.58	17.58	17.25	17.25	17.25	17.25	17.22	17.22	17.21	17.21	17.2	17.17
ORNoC	17.59	17.59	17.59	17.58	17.58	17.25	17.25	17.26	17.25	17.22	17.21	17.21	17.21	17.21	17.18

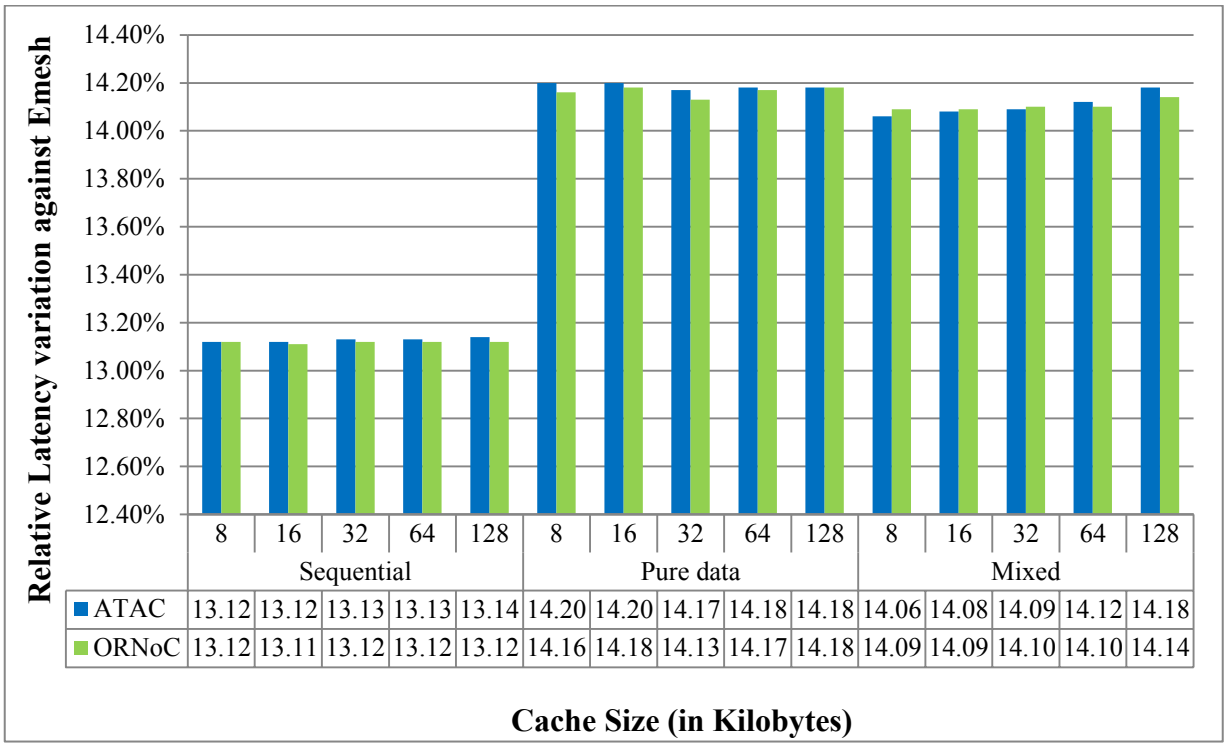
Figure 97: Average Packet Latency for Cache Sizes N = {8, 16, 32, 64, 128}

The average packet latency shows only very slight variations the same for increasing cache sizes which shows that the varying cache sizes do not significantly affect the packet latency in the network. For the Emesh architecture, the cache values are observed to decrease from 20.25ns for 8KB to 20.24ns for 128KB, 20.1ns to 20.07ns for the pure data parallelism implementation and 20.04ns to 20.01ns for the mixed parallelism implementation. The ATAC architecture shows values of 17.59ns for 8KB and decreases to17.58ns for 128KB cache for sequential implementation, 17.25ns to 17.22ns for pure data parallelism implementation and 17.22ns to 17.17ns for mixed parallelism implementation. The ORNoC architecture shows similar values as that of ATAC for both sequential and pure data

parallelism implementation, while the values are 17.21ns for 8KB cache and reduces to 17.18ns for 128KB for the mixed parallelism implementation.

The average values for packet latency for sequential implementation are 20.245ns for Emesh and 17.586ns for both ATAC and ORNoC architectures for all the cache sizes considered. For all the cache sizes considered, the average values are 20.094ns for Emesh, 17.244ns for ATAC and 17.246ns for ORNoC architectures in the case of pure data parallelism implementation. The mixed implementation gives average values of 20.032ns, 17.202ns and 17.204ns for Emesh, ATAC and ORNoC architectures respectively for all the cache sizes considered. The average improvement for the pure data parallelism and mixed parallelism implementations against sequential implementation are 0.75% and 1.05% for Emesh, while it is 1.94% and 2.18% for ATAC and 1.93% and 2.17% for ORNoC for all the cache sizes considered.

The observations of the results show that for this set of experiments, minimum number of packets is sent in the network by the mixed parallelism implementation scheme, followed by the pure data parallelism implementation and the largest number of packets is sent by sequential implementation, and hence packet latency values among the implementations also increase in the same order.



	8	16	32	64	128	8	16	32	64	128	8	16	32	64	128
ATAC	13.12	13.12	13.13	13.13	13.14	14.20	14.20	14.17	14.18	14.18	14.06	14.08	14.09	14.12	14.18
ORNoC	13.12	13.11	13.12	13.12	13.12	14.16	14.18	14.13	14.17	14.18	14.09	14.09	14.10	14.10	14.14

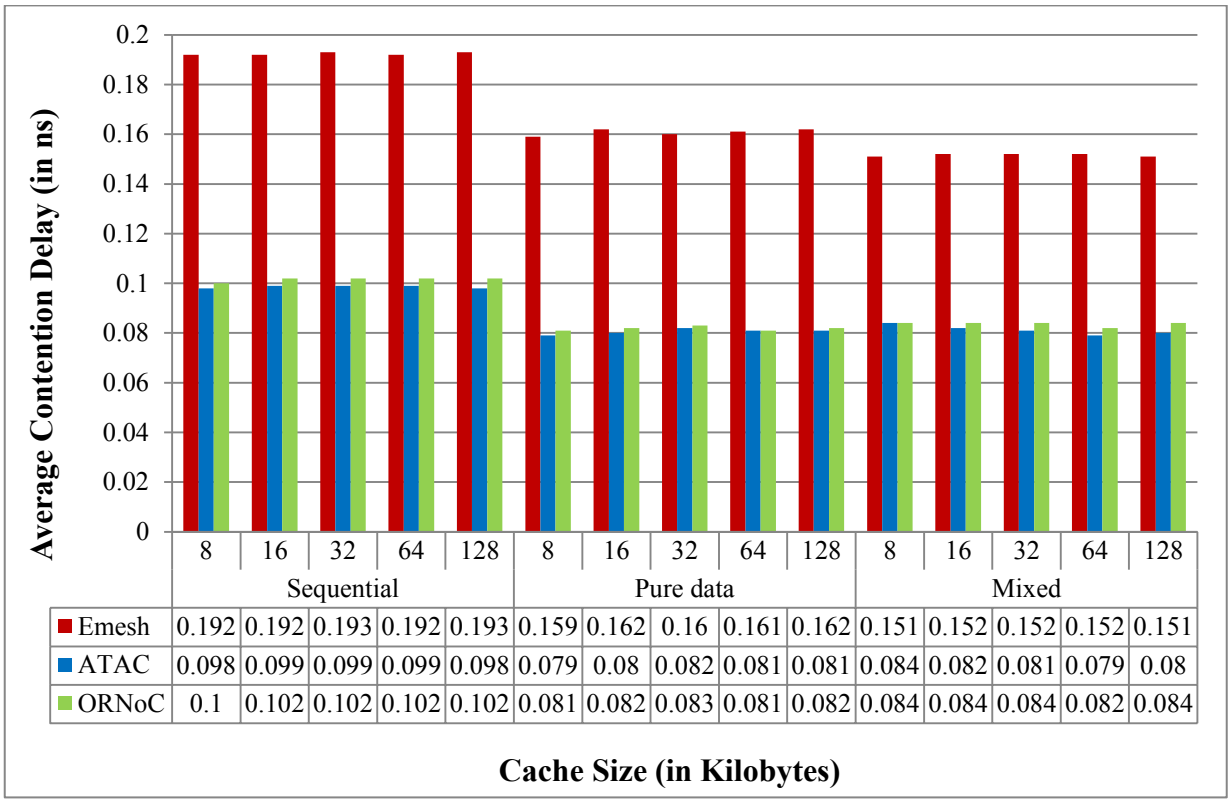
Figure 98: Relative Latency variation for Cache Sizes N = {8, 16, 32, 64, 128}

The relative variation in packet latency values for ATAC architecture are 13.12% for a cache size of 8KB and increases slightly to 13.14% for the cache size of 128KB in the case of sequential implementation, goes from 14.2% for 8KB cache to 14.18% for 128KB cache in the case of pure data implementation and 14.06% for the cache size of 8KB to 14.18% in the case of 128KB cache for mixed parallelism implementation. The relative variation in packet latency values for ORNoC architecture are 13.12% for all the cache sizes except for 16KB, which has 13.11% in the case of sequential implementation, goes from 14.16% for 8KB cache to 14.18% for 128KB cache in the case of pure data implementation and 14.09% for the cache size of 8KB to 14.14% in the case of 128KB cache for mixed parallelism implementation.

The relative variation in packet latency has average values of 13.13% and 13.12% for ATAC and ORNoC respectively in the case of sequential implementation, 14.19% for ATAC and

14.16% for ORNoC in the case of pure data parallelism implementation and 14.11% for ATAC and 14.10% for ORNoC in the case of mixed parallelism implementation for all the cache sizes considered. Both the ONoC architectures have almost the same percentage variations in latency with respect to Emesh for each of the implementations. The ONoC architectures have better packet latency values than Emesh due to the presence of optical network for inter cluster communication.

5.8.2 Results for Contention Delay for varying Size of L1-Data Cache



	8	16	32	64	128	8	16	32	64	128	8	16	32	64	128
	Sequential					Pure data					Mixed				
Emesh	0.192	0.192	0.193	0.192	0.193	0.159	0.162	0.16	0.161	0.162	0.151	0.152	0.152	0.152	0.151
ATAC	0.098	0.099	0.099	0.099	0.098	0.079	0.08	0.082	0.081	0.081	0.084	0.082	0.081	0.079	0.08
ORNoC	0.1	0.102	0.102	0.102	0.102	0.081	0.082	0.083	0.081	0.082	0.084	0.084	0.084	0.082	0.084

Figure 99: Average Contention Delay for Cache Sizes N = {8, 16, 32, 64, 128}

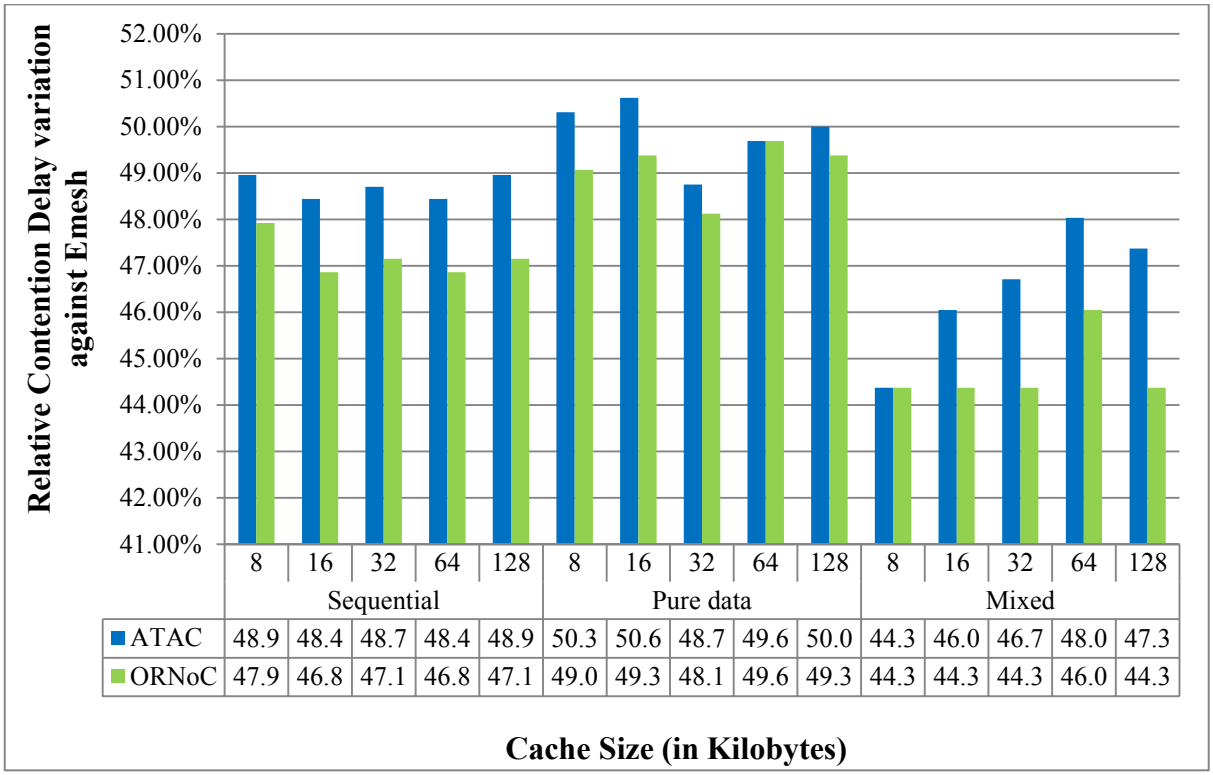
The contention delay values do not show much change with increase in cache sizes for each of the architectures. For the Emesh architecture, the contention delay values are observed to increase slightly from 0.192ns for 8KB to 0.193ns for 128KB, 0.159ns for 8KB cache to

0.162ns for 128KB cache in the case of pure data parallelism implementation and from 0.151ns for 8KB slightly increases to 0.152ns for intermediate cache sizes and goes back to 0.151ns for 128KB cache in the case of mixed parallelism implementation. The ATAC architecture shows values of 17.59ns for 8KB and decreases to17.58ns for 128KB cache for sequential implementation, 0.079ns for 8KB cache size to a maximum value of 0.082ns for 32KB cache and slightly varies to 0.081ns with increased cache sizes for pure data parallelism implementation and 0.084ns for a cache size of 8KB to 0.08ns for a cache size of 128KB for mixed parallelism implementation. The ORNoC architecture shows the contention delay values of 0.1ns for 8KB cache which marginally increases to 0.102ns for 128KB for the sequential implementation, increases form 0.081ns for 8KB cache to 0.083ns for 64 KB cache and then decreases again to 0.081ns for the pure data parallelism implementation and closely following ATAC, the mixed parallelism implementation has contention delay values for ORNoC that are uniformly 0.084ns with the exception of 64KB which has 0.082ns.

The average values of contention delay for sequential implementation are 0.192ns, 0.099ns and 0.102ns for Emesh, ATAC and ORNoC architectures respectively for all the cache sizes considered. Pure data parallelism implementation gives average values of 0.161ns for Emesh, 0.081ns for ATAC and 0.082ns for ORNoC architectures for all the cache sizes. The average values for mixed implementation are 0.152ns, 0.081ns and 0.084ns respectively for Emesh, ATAC and ORNoC architectures respectively for all the cache sizes.

The average variation for all the cache sizes between the pure data parallelism and mixed parallelism against sequential implementation are 16.15% and 20.83% for Emesh, 18.18% for both implementations in ATAC and 19.61% and 17.65% for ORNoC. The number of

packets transferred in the network is greater for the sequential implementation as compared to both the parallelized implementations and hence, sequential implementation has greater contention delay.



	Sequential					Pure data					Mixed				
	8	16	32	64	128	8	16	32	64	128	8	16	32	64	128
ATAC	48.9	48.4	48.7	48.4	48.9	50.3	50.6	48.7	49.6	50.0	44.3	46.0	46.7	48.0	47.3
ORNoC	47.9	46.8	47.1	46.8	47.1	49.0	49.3	48.1	49.6	49.3	44.3	44.3	44.3	46.0	44.3

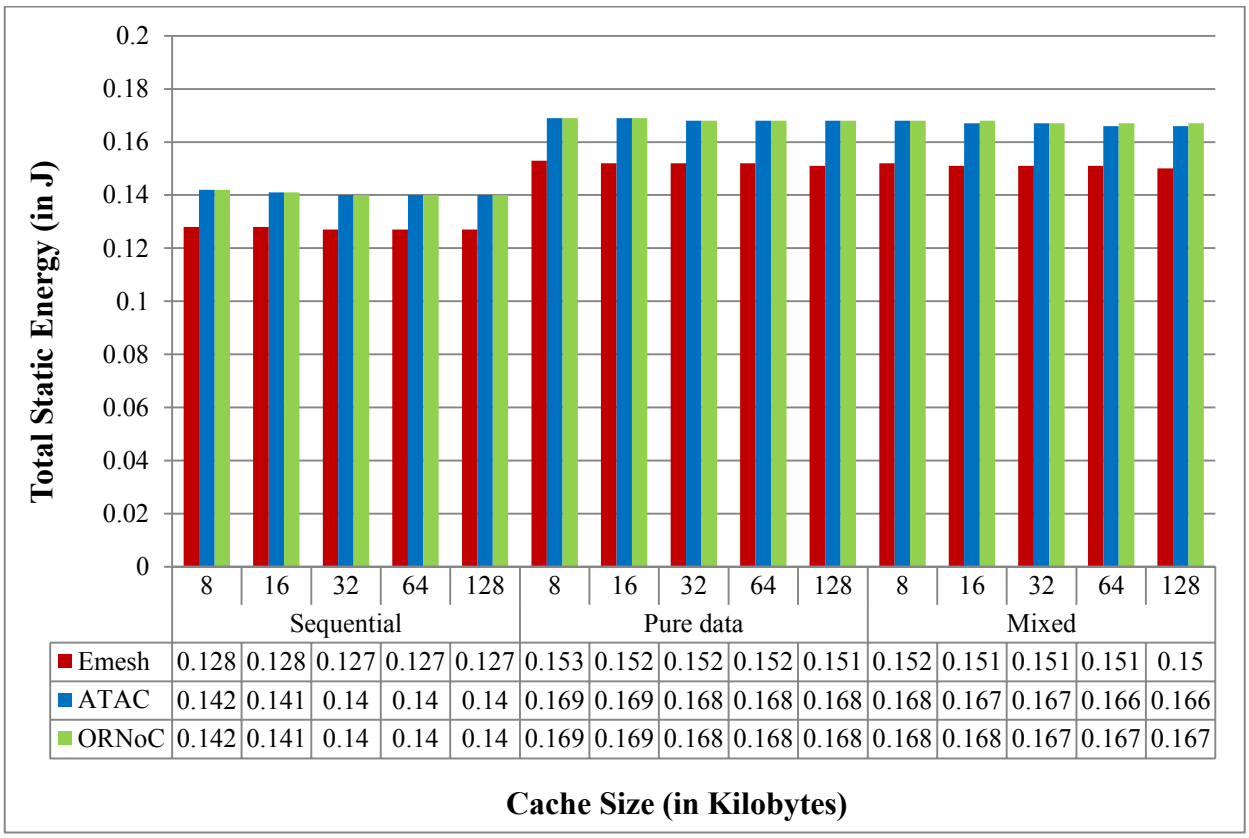
Figure 100: Relative Contention Delay variation for Cache Sizes N = {8, 16, 32, 64, 128}

The relative variation in contention delay values for ATAC architecture are 48.9% for a cache size of 8KB and varies to 48.4% and 48.7% for intermediate cache sizes (16KB,64KB and 32 KB respectively) and goes back to 48.9% for the cache size of 128KB in the case of sequential implementation, goes from 50.3% for 8KB cache to 50% for 128KB cache in the case of pure data implementation and 44.3% for the cache size of 8KB to 47.3% in the case of 128KB cache for mixed parallelism implementation. The relative variation in contention delay values for ORNoC architecture are 47.9% and goes down slightly to 47.1% for 128KB cache size in the case of sequential implementation, goes from 49% for 8KB cache to 49.3%

for 128KB cache in the case of pure data implementation and 44.3% for all the cache sizes except 64KB which has 46% improvement for mixed parallelism implementation.

The average improvement for all the cache sizes in the contention delay values of the ONoC architectures against Emesh are: 48.66% for ATAC and 47.14% for ORNoC in the case of sequential implementation, 49.84% and 49.06% for ATAC and ORNoC respectively in the case of pure data parallelism implementation, 46.46% for ATAC and 44.64% for ORNoC in the mixed parallelism implementation. The ATAC and ORNoC values are observed to be very close to each other when compared in terms of relative variation against Emesh. Emesh architecture shows much higher delays as it only has electrical networks for communication, while both the ONoC architectures have better contention delay values than Emesh. The contention delay values for both the ONoC architectures only have very little difference.

5.8.3 Results for Static Energy for varying Size of L1-Data Cache



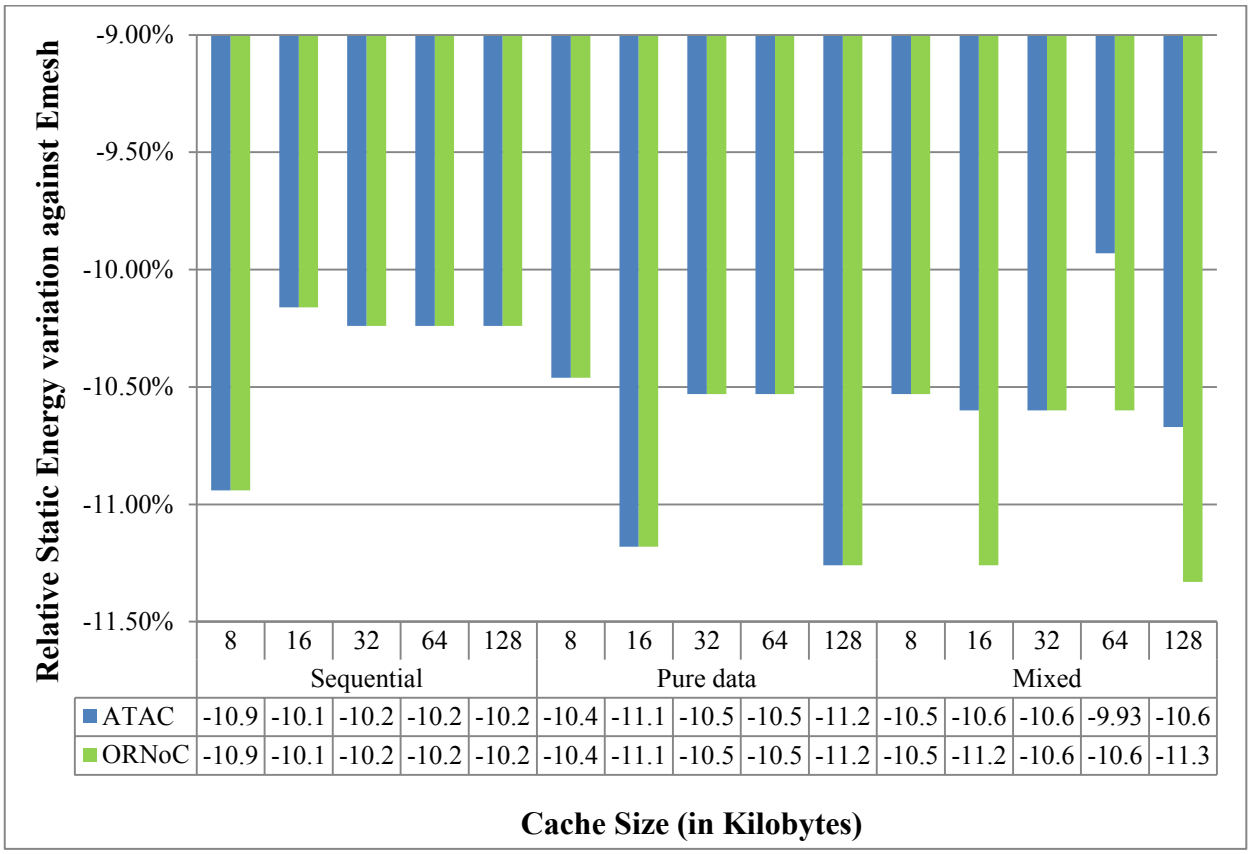
	8	16	32	64	128	8	16	32	64	128	8	16	32	64	128
	Sequential					Pure data					Mixed				
Emesh	0.128	0.128	0.127	0.127	0.127	0.153	0.152	0.152	0.152	0.151	0.152	0.151	0.151	0.151	0.15
ATAC	0.142	0.141	0.14	0.14	0.14	0.169	0.169	0.168	0.168	0.168	0.168	0.167	0.167	0.166	0.166
ORNoC	0.142	0.141	0.14	0.14	0.14	0.169	0.169	0.168	0.168	0.168	0.168	0.168	0.167	0.167	0.167

Figure 101: Static Energy consumption for Cache Sizes N = {8, 16, 32, 64, 128}

The static energy consumption is observed to remain almost uniform for each specific implementation with increasing cache sizes, which indicates that the variation in cache size does not contribute much to the non-data dependent components in the network. For the Emesh architecture, the static energy values decrease from 0.128J for 8KB to 0.127J for 128KB, 0.153J to 0.151J for the pure data parallelism implementation and 0.152J to 0.15J for the mixed parallelism implementation. The ATAC architecture shows values of 0.142J for 8KB and decreases to0.14Jfor 128KB cache for sequential implementation, 0.169J for cache size of 8KB to 0.168J for cache size of 128KB for pure data parallelism implementation and 0.168J for 8KB cache to 0.166J for 128KB cache for mixed parallelism

implementation. The ORNoC architecture shows similar values as that of ATAC for both sequential and pure data parallelism implementation, while the values are 0.168J for 8KB cache and reduces to 0.167J for 128KB for the mixed parallelism implementation.

The average values of contention delay for all the cache sizes for sequential implementation are 0.127J for Emesh and 0.141J for both ATAC and ORNoC architectures. Pure data parallelism implementation gives average values of 0.152J for Emesh and 0.168J for both ATAC and ORNoC architectures for all the cache sizes. The average values for mixed implementation are 0.151J for Emesh and 0.167J for both ATAC and ORNoC architectures for all the cache sizes under consideration. The average variation for all the cache sizes between the pure data parallelism and mixed parallelism against sequential implementation are -19.68% and -18.90% for Emesh, -19.15% and -18.44% for both ATAC and ORNoC architectures. Both the parallelized implementations efficiently use more number of packets in the network, which leads to the usage of more data independent components in the network and in turn increases the static energy consumption as compared to sequential implementation.



	8	16	32	64	128	8	16	32	64	128	8	16	32	64	128
	Sequential					Pure data					Mixed				
ATAC	-10.9	-10.1	-10.2	-10.2	-10.2	-10.4	-11.1	-10.5	-10.5	-11.2	-10.5	-10.6	-10.6	-9.93	-10.6
ORNoC	-10.9	-10.1	-10.2	-10.2	-10.2	-10.4	-11.1	-10.5	-10.5	-11.2	-10.5	-11.2	-10.6	-10.6	-11.3

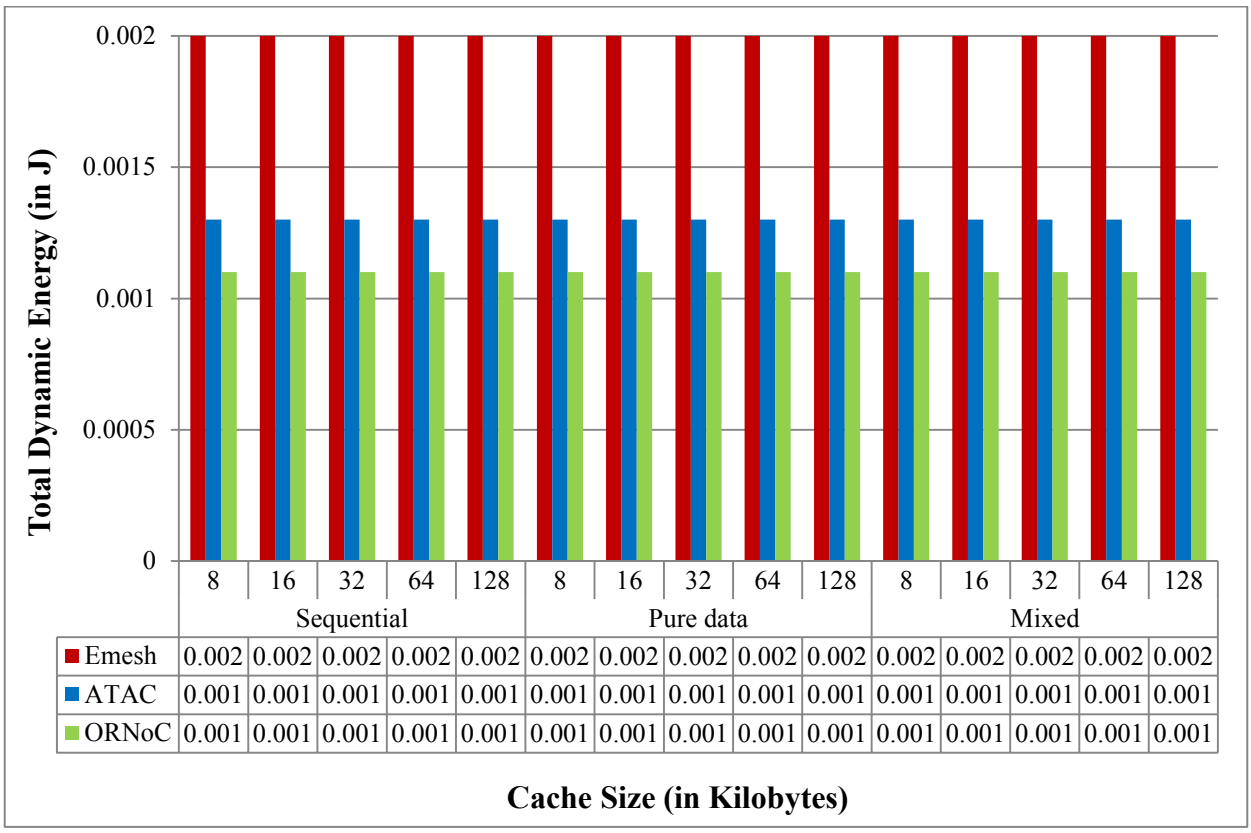
Figure 102: Relative Static Energy variation for Cache Sizes N = {8, 16, 32, 64, 128}

The ONoCs show a decrease in the relative static energy consumption as compared to Emesh which is indicated by the negative values. Since the actual values remain almost uniform, as clear from the previous figure, the relative variation also remains very close to each other for all the three implementations. The relative variation in static energy consumption values for ATAC architecture are -10.9% for a cache size of 8KB and goes to -10.2% for the cache size of 128KB in the case of sequential implementation, goes from -10.4% for 8KB cache to -11.2% for 128KB cache in the case of pure data implementation and -10.5% for the cache size of 8KB to -10.6% in the case of 128KB cache for mixed parallelism implementation. The relative variation in static energy consumption values for ORNoC architecture are similar to that of ATAC for both sequential and pure data

parallelism implementation schemes and -10.5% for 8KB cache size and goes to -11.2% for 128KB cache size in the case of mixed implementation scheme.

The average improvement rate in the static energy values of the ONoC architectures against Emesh for all the cache sizes are: -10.3% for both ATAC and ORNoC in the case of sequential implementation, -10.74% for both ATAC and ORNoC in the case of pure data parallelism implementation, - 10.45% for ATAC and -10.84% for ORNoC in the mixed parallelism implementation. The static energy consumption values for Emesh are observed to be lower as compared to the ONoC architectures because of the additional optical components in ONoCs [54]. Among the optical networks, ORNoC and ATAC architectures have similar values for static energy consumption as the cache size parameter does not cause significant difference in the number of optical components.

5.8.4 Results for Dynamic Energy for varying Size of L1-Data Cache



	8	16	32	64	128	8	16	32	64	128	8	16	32	64	128
	Sequential					Pure data					Mixed				
Emesh	0.002	0.002	0.002	0.002	0.002	0.002	0.002	0.002	0.002	0.002	0.002	0.002	0.002	0.002	0.002
ATAC	0.001	0.001	0.001	0.001	0.001	0.001	0.001	0.001	0.001	0.001	0.001	0.001	0.001	0.001	0.001
ORNoC	0.001	0.001	0.001	0.001	0.001	0.001	0.001	0.001	0.001	0.001	0.001	0.001	0.001	0.001	0.001

Figure 103: Dynamic Energy consumption for Cache Sizes N = {8, 16, 32, 64, 128}

The dynamic energy consumption values show no variation with increase in cache size for any of the architectures in any of the implementations, because it is predominantly data dependent energy and the workload does not vary for this experiment. The dynamic energy consumption values remain constant at 0.002J for Emesh architecture and 0.001J for both ATAC and ORNoC architectures for all the cache sizes under consideration. The average values for all the cache sizes are observed to be to 0.002J for Emesh architecture, 0.0013J for ATAC and 0.0011J for ORNoC architectures for all the implementations. Since the values are uniform throughout, there are no relative variations with respect to the sequential implementation in the other two schemes. The dynamic energy consumption values are

observed to be greater for Emesh as compared to both the ONoC architectures. The different implementation schemes for each of the architectures have similar dynamic energy consumption as they work on the same amount of data, which constitutes the dynamic energy consumption.

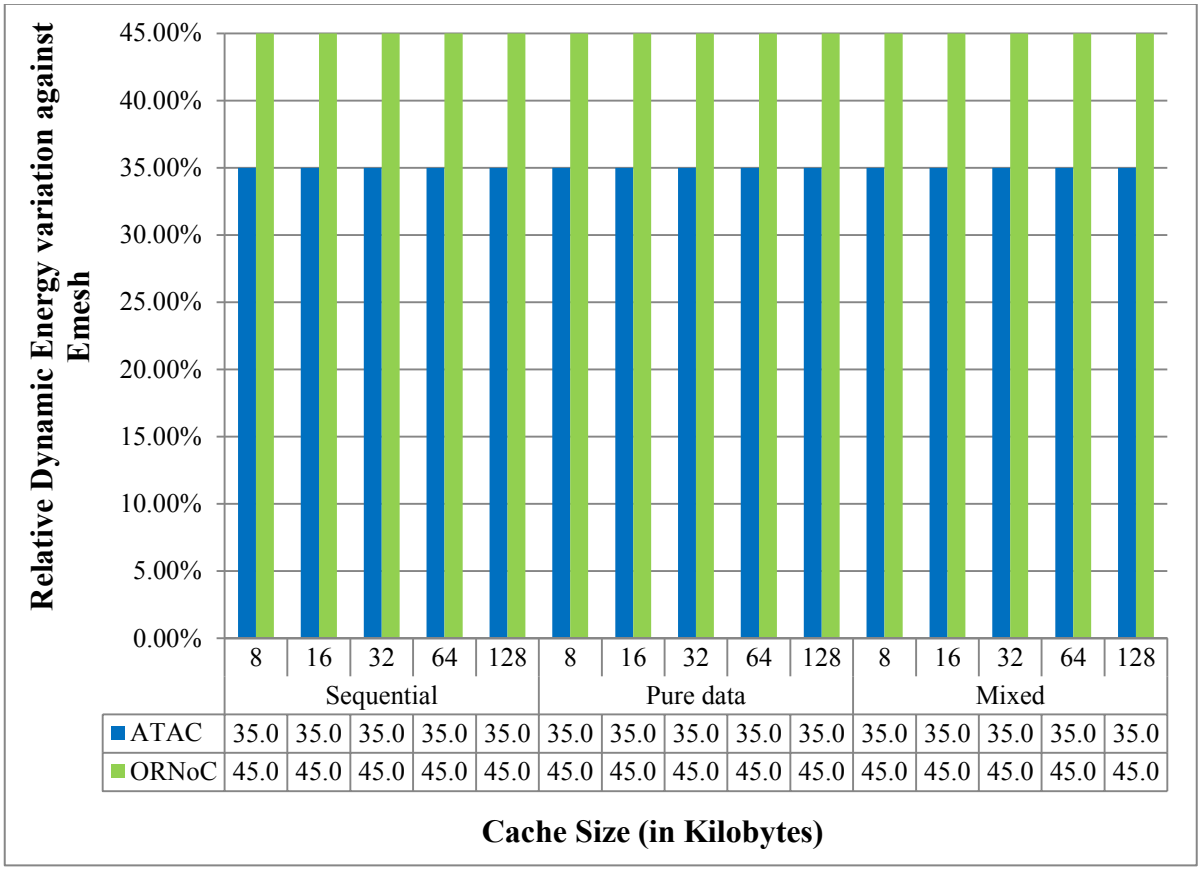
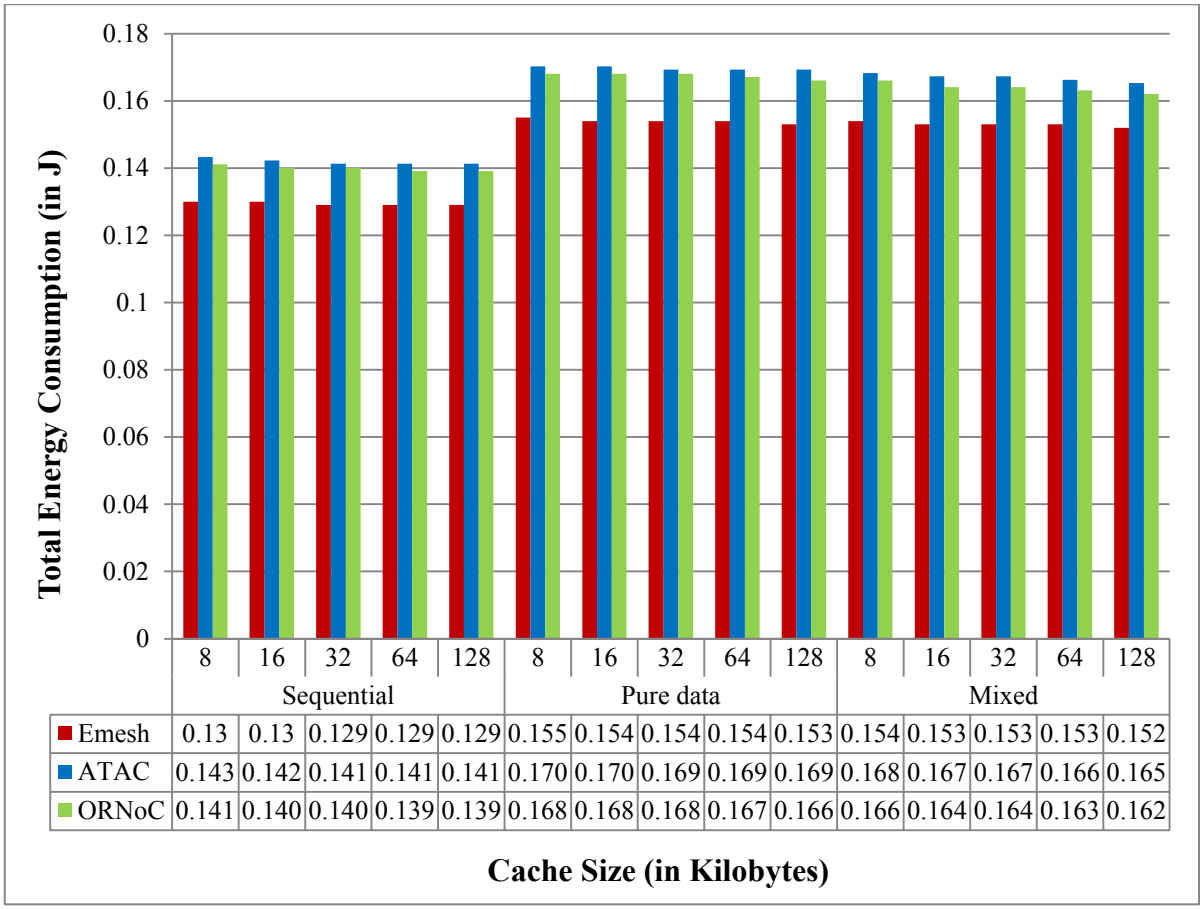


Figure 104: Relative Dynamic Energy variation for Cache Sizes N = {8, 16, 32, 64, 128}

As mentioned above, since the actual values of dynamic energy remains the same for all the architectures for all the implementations, the actual and average variations also remain uniform for both ONoC architectures for all the implementation schemes. ORNoC is observed to be better with 45% improvement as compared to Emesh, while ATAC shows an improvement of 35% as clear from the figure. Emesh architecture has more electrical links that contribute to the data dependent energy and hence, it dynamic energy consumption is

greater for Emesh architecture as compared to ONoC architectures [54]. ORNoC architecture uses lesser number of wavelengths and waveguides which in turn lead to decreased number of modulators and receivers that contribute to data dependent energy consumption and hence, has slightly better dynamic energy consumption as compared to ATAC [54].

5.8.5 Results for Total Energy for varying Size of L1-Data Cache

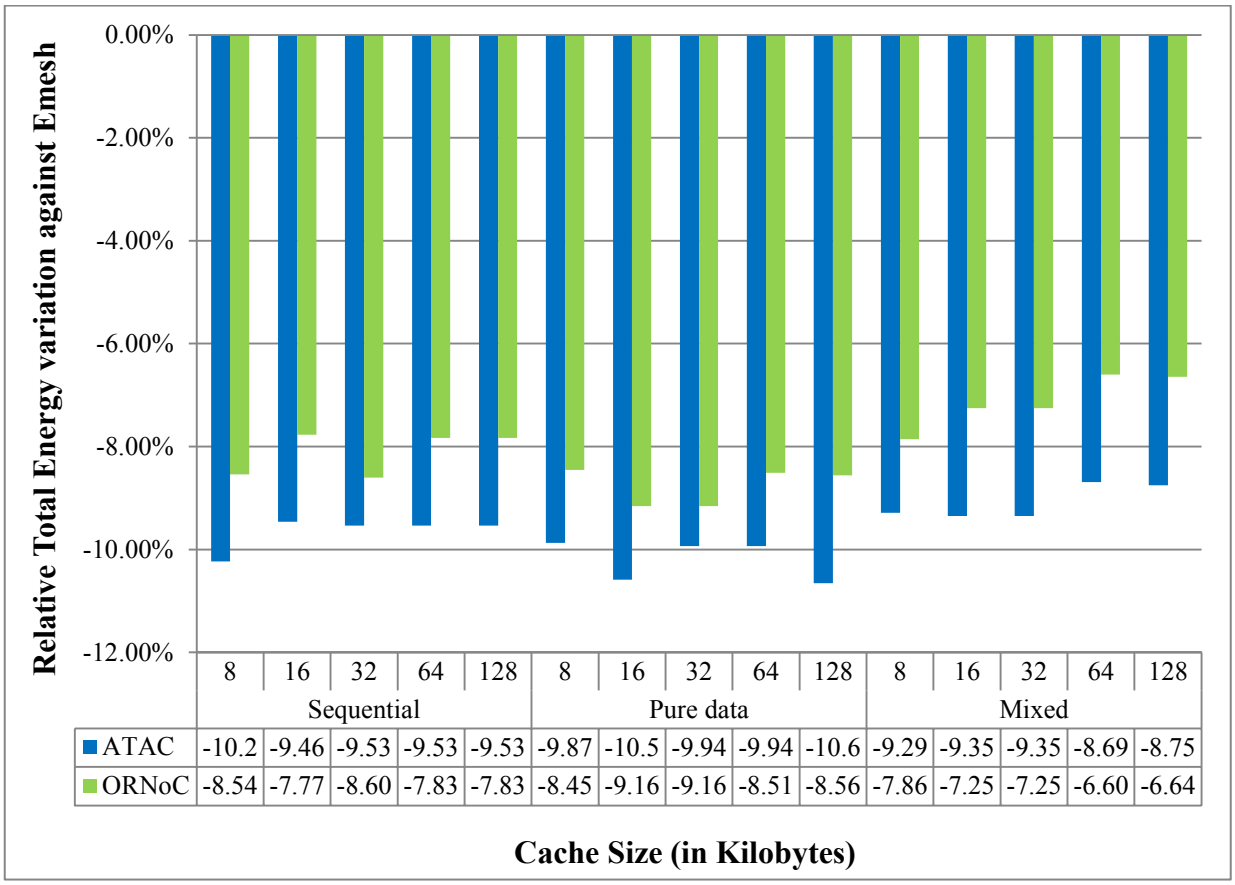


	8	16	32	64	128	8	16	32	64	128	8	16	32	64	128
	Sequential					Pure data					Mixed				
Emesh	0.13	0.13	0.129	0.129	0.129	0.155	0.154	0.154	0.154	0.153	0.154	0.153	0.153	0.153	0.152
ATAC	0.143	0.142	0.141	0.141	0.141	0.170	0.170	0.169	0.169	0.169	0.168	0.167	0.167	0.166	0.165
ORNoC	0.141	0.140	0.140	0.139	0.139	0.168	0.168	0.168	0.167	0.166	0.166	0.164	0.164	0.163	0.162

Figure 105: Total Energy consumption for Cache Sizes N = {8, 16, 32, 64, 128}The static energy consumption forms the major contributing factor in the total energy consumption, and the dynamic energy consumption forms a very small fraction compared to it and also remains uniform for all the implementations for all the architectures. Hence, the trend for total energy consumption resembles the static energy consumption graph, with the Emesh

having an advantage over the ONoC architectures with similar energy consumption profile due to the same reasons as in the case of static energy consumption. For the Emesh architecture, the total energy consumption values decrease from 0.13J for 8KB to 0.129J for 128KB, 0.155J to 0.153J for the pure data parallelism implementation and 0.154J to 0.152J for the mixed parallelism implementation. The ATAC architecture shows values of 0.143J for 8KB and decreases to0.141Jfor 128KB cache for sequential implementation, 0.17J for a cache size of 8KB to 0.169J for a cache size of 128KB for pure data parallelism implementation and 0.168J for 8KB cache to 0.165J for 128KB cache in the case of mixed parallelism implementation. The ORNoC architecture shows values of 0.141J for 8KB and decreases to0.139J for 128KB cache for sequential implementation, 0.168J for a cache size of 8KB to 0.166J for a cache size of 128KB for pure data parallelism implementation and 0.166J for 8KB cache to 0.162J for 128KB cache in the case of mixed parallelism implementation.

The average values of total energy consumption are calculated for all the cache sizes. For sequential implementation are 0.129J for Emesh, 0.142J for ATAC and 0.140J for ORNoC architectures. Pure data parallelism implementation gives average values of 0.154J for Emesh and 0.169J for ATAC and 0.168J for ORNoC architectures. The average values for mixed implementation are 0.153J for Emesh and 0.167J for ATAC and 0.164J for ORNoC architectures. The average variation between the pure data parallelism and mixed parallelism against sequential implementation are -19.38% and -18.60% for Emesh, -19.01% and -17.60% for ATAC and -20% and -17.14% for ORNoC architectures.



	8	16	32	64	128	8	16	32	64	128	8	16	32	64	128
	Sequential					Pure data					Mixed				
ATAC	-10.2	-9.46	-9.53	-9.53	-9.53	-9.87	-10.5	-9.94	-9.94	-10.6	-9.29	-9.35	-9.35	-8.69	-8.75
ORNoC	-8.54	-7.77	-8.60	-7.83	-7.83	-8.45	-9.16	-9.16	-8.51	-8.56	-7.86	-7.25	-7.25	-6.60	-6.64

Figure 106: Relative Total Energy variation for Cache Sizes N = {8, 16, 32, 64, 128}

The relative total energy variation shows a decreasing trend among the ONoC architectures in comparison with Emesh. The relative variation in total energy consumption values for ATAC architecture are -10.2% for a cache size of 8KB and goes to -9.53% for the cache size of 128KB in the case of sequential implementation, goes from -9.87% for 8KB cache to -10.6% for 128KB cache in the case of pure data implementation and -9.29% for the cache size of 8KB to -8.75% in the case of 128KB cache for mixed parallelism implementation. The relative variation in total energy consumption values for ORNoC architecture are -8.54% for a cache size of 8KB and goes to -7.83% for the cache size of 128KB in the case of sequential implementation, goes from -8.45% for 8KB cache to -8.56% for 128KB cache in the case of pure data implementation and -7.86% for 8KB cache size and goes to -6.64% for

128KB cache size in the case of mixed implementation scheme. The average degradation in the contention delay values for all the cache sizes for the ONoC architectures against Emesh are: -9.65% for ATAC and -8.11% for ORNoC in the case of sequential implementation, -10.17% and -8.77% for ATAC and ORNoC respectively in the case of pure data parallelism implementation, -9.09% for ATAC and -7.12% for ORNoC in the mixed parallelism implementation.

Chapter 6

Conclusion

This thesis compares the Emesh, ATAC and ORNoC NoC architectures, for the network latency, contention delay and network energy, in terms of three different software parallelized versions of the Canny edge detection algorithm. The simulations were performed using the Graphite NoC simulator and utilize different sets of configuration parameters to explore different design alternatives and observe their influence on the network latency, contention delay and network energy. Primary importance was given to the impact of different parallelization strategies on the implementation of the application. The Caany edge application implementations using the sequential scheme, the pure data parallelization scheme and the mixed parallelization scheme were performed and tested in the native machine before being used for comparison of the NoC architectures. In terms of the average packet latency and contention delay, the mixed parallelization scheme outperforms the other parallelization schemes in most of the experiments. The sequential or serialized implementation scheme provides better static energy results compared to its parallelized counterparts, and furthermore the results for Emesh show that they consume lesser energy as compared to the ONoC architectures.

The simulations provide results to evaluate the optimal configuration parameter for the Canny edge detection algorithm for the workload under consideration. For the image of 1024x1024 pixels, 64 cores are observed to give optimal results for latency and contention delay. The optimal cluster size is observed to be 16 for all the implementations.. Greater

value of flit width and number of optical access points minimize latency and contention delay and hence, 64 bits and 8 are optimal values respectively for each of them for the corresponding workload considered. If the increase in energy consumption is a concern, the values for all the parameters used should be lesser, with some trade-off between delay and energy consumption. For example, the number of cores that gives optimum latency and contention delay is 64 and not 16 or 256. The routing strategies do not differentiate much in any of the parameters and hence, any of them can be used for the workload considered. The best energy consumption among the different cache line sizes is obtained for 32 bytes, while 16 bytes offer slightly better latency and contention delay results than 32 bytes. The cache size does not significantly affect any of the outputs for the observed workloads and hence, it can be decided by the designer based on other parameters like the area and cost of manufacturing.

Hence, the results from the experiments make it clear that the type of parallelization scheme adapted can impact the performance of the network even in the optical network on chip architectures. Also, it can be concluded that the Emesh network with the sequential style of coding is only preferable if the energy consumption is to be reduced, which primarily accounts for the energy consumed by the optical components in the ONoC architectures, but are still highly disadvantageous in the case of larger networks. If the priority is lesser contention delay and latency in the network it is advisable to go for the mixed parallelization scheme in the ORNoC architecture it provides better latency and contention delay and also consumes lesser power compared to the ATAC architecture.

Chapter 7

Future Work

The current implementation of the Canny edge detection application involves the edge detection performed on a single image of different sizes. The results of this experimental application-based evaluation indicate the advantages of the optical networks over the electrical networks with respect to the latency and contention delay. A better evaluation of the network can be obtained when the edge detection is performed on a stream of images, which can lead to increased traffic in the communication network and the memory management schemes. Medical imaging and computer vision fields involve many video streaming applications which require fast processing of multiple image frames of high quality in an accurate and time bound which can be processed much faster with multicore networks utilizing optical networks on chip for communication between the IP modules. The image processing performed in this thesis can be extended in the future to video processing applications for real-time applications so as to implement processing with complete hardware and software parallelization. Also, real time video processing is another extension for the current application for evaluation of the different on chip network architectures.

References

[1] "CUDA | GeForce." [Online]. Available: https://www.geforce.com/hardware/technology/cuda. [Accessed: 09-Oct-2017].

[2] "Intel® Threading Building Blocks (Intel® TBB) | Intel® Software." [Online]. Available: https://software.intel.com/en-us/intel-tbb. [Accessed: 09-Oct-2017].

[3] A. Shacham, K. Bergman, and L. P. Carloni, "On the Design of a Photonic Network-on-Chip."

[4] T. L. Ben Cheikh, G. Beltrame, G. Nicolescu, F. Cheriet, and S. Tahar, "Parallelization strategies of the canny edge detector for multi-core CPUs and many-core GPUs," *2012 IEEE 10th Int. New Circuits Syst. Conf. NEWCAS 2012*, vol. 1, pp. 49–52, 2012.

[5] J. P. Singh, W.-D. Weber, and A. Gupta, "SPLASH: STANFORD PARALLEL APPLICATIONS FOR SHARED-MEMORY," 1991.

[6] C. Bienia, S. Kumar, J. P. Singh, and K. Li, "The PARSEC benchmark suite," in *Proceedings of the 17th international conference on Parallel architectures and compilation techniques - PACT '08*, 2008, p. 72.

[7] "Edge Detection - MATLAB & Simulink." [Online]. Available: https://www.mathworks.com/discovery/edge-detection.html. [Accessed: 08-Oct-2017].

[8] J. Canny, "A Computational Approach to Edge Detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, no. 6, 1986.

[9] J. E. Miller, H. Kasture, G. Kurian, C. Gruenwald, N. Beckmann, C. Celio, J. Eastep, and A. Agarwal, "Graphite: A distributed parallel simulator for multicores," in *HPCA*

- *16 2010 The Sixteenth International Symposium on High-Performance Computer Architecture*, 2010, no. January, pp. 1–12.

[10] L. Benini, G. De Micheli, L. Benini, and G. De Micheli, "Chapter 2 – Network Architecture: Principles and Examples," in *Networks on Chips*, 2006, pp. 23–43.

[11] M. Miti, M. Stojč, and Z. Stamenkovi, "An Overview of SoC Buses," in *Digital Systems and Applications*, vol. 11, no. 7, 2007, pp. 1–17.

[12] S. Le Beux, J. Trajkovic, I. O 'connor, G. Nicolescu, G. Bois, and P. Paulin, "Optical Ring Network-on-Chip (ORNoC): Architecture and Design Methodology," 2011.

[13] A. Shacham, B. G. Lee, A. Biberman, K. Bergman, and L. P. Carloni, "Photonic NoC for DMA communications in chip multiprocessors," *Proc. - 15th Annu. IEEE Symp. High-Performance Interconnects, HOT Interconnects*, no. October 2016, pp. 29–36, 2007.

[14] D. Vantrease, R. Schreiber, M. Monchiero, M. Mclaren, N. P. Jouppi, M. Fiorentino, A. Davis, N. Binkert, R. G. Beausoleil, and J. H. Ahn, "Corona: System Implications of Emerging Nanophotonic Technology."

[15] S. Le Beux, H. Li, I. O'Connor, K. Cheshmi, X. Liu, J. Trajkovic, and G. Nicolescu, "Chameleon: Channel efficient Optical Network-on-Chip," vol. 2014, pp. 1–6, 2014.

[16] Y. Pan, P. Kumar, J. Kim, G. Memik, Y. Zhang, and A. Choudhary, "Firefly : Illuminating Future Network-on-Chip with Nanophotonics Categories and Subject Descriptors," pp. 429–440.

[17] G. Kurian, J. E. Miller, J. Psota, J. Eastep, J. Liu, J. Michel, L. C. Kimerling, and A. Agarwal, "ATAC: A 1000-Core Cache-Coherent Processor with On-Chip Optical Network," 2010.

[18] A. Shacham, K. Bergman, and L. P. Carloni, “Photonic networks-on-chip for future generations of chip multiprocessors,” *IEEE Trans. Comput.*, vol. 57, no. 9, pp. 1246–1260, Sep. 2008.

[19] S. Pasricha and N. Dutt, “2008-ORB_ an on-chip optical ring bus communication architecture for multi-processor systems-on-chip.pdf,” pp. 789–794, 2008.

[20] N. Kirman, M. Kirman, R. K. Dokania, J. F. Martínez, A. B. Apsel, M. A. Watkins, and D. H. Albonesi, “Leveraging optical technology in future bus-based chip multiprocessors,” *Proc. Annu. Int. Symp. Microarchitecture, MICRO*, pp. 492–503, 2006.

[21] Y. Pan, J. Kim, and G. Memik, “FlexiShare: Channel Sharing for an Energy-Efficient Nanophotonic Crossbar,” *Hpca’10*, pp. 1–12, 2010.

[22] S. Le Beux, H. Li, I. O ’connor, K. Cheshmi, X. Liu, J. Trajkovic, and G. Nicolescu, “CHAMELEON: CHANNEL Efficient Optical Network-on-Chip.”

[23] M. Jones, “NoCsim : a versatile network on chip simulator,” 2005.

[24] A. B. Kahng, B. Li, and L. Peh, “ORION 2 . 0 : A Power-Area Simulator for Interconnection Networks,” *Tvlsi*, vol. XX, no. 1, pp. 1–5, 2010.

[25] S. Rumley, M. Bahadori, K. Wen, D. Nikolova, and K. Bergman, “PhoenixSim: Crosslayer Design and Modeling of Silicon Photonic Interconnects,” p. 7:1–7:6, 2016.

[26] C. Sun, C. H. O. Chen, G. Kurian, L. Wei, J. Miller, A. Agarwal, L. S. Peh, and V. Stojanovic, “DSENT - A tool connecting emerging photonics with electronics for opto-electronic networks-on-chip modeling,” *Proc. 2012 6th IEEE/ACM Int. Symp. Networks-on-Chip, NoCS 2012*, pp. 201–210, 2012.

[27] J. Nickolls, I. Buck, M. Garland, and K. Skadron, “Scalable parallel programming

with CUDA," *Queue*, no. April, p. 1, 2008.

[28] K. Ogawa, Y. Ito, and K. Nakano, "Efficient Canny Edge Detection Using a GPU," *2010 First Int. Conf. Netw. Comput.*, pp. 279–280, 2010.

[29] "Canny edge detector - Rosetta Code." [Online]. Available: https://rosettacode.org/wiki/Canny_edge_detector. [Accessed: 02-Nov-2017].

[30] M. D. Hill and M. R. Marty, "Amdahl's Law in the Multicore Era," 2007.

[31] G. E.Moore, "Cramming more components onto integrated circuits," *Electronics*, vol. 38, no. 8, 1965.

[32] C. Kessler and J. Keller, "Models for Parallel Computing: Review and Perspectives," in *PARS-Mitteilugen*, 2007, no. 24, pp. 13–29.

[33] L. Benini, G. De Micheli, L. Benini, and G. De Micheli, "Chapter 1 – Networks on Chip," in *Networks on Chips*, 2006, pp. 1–22.

[34] M. Mitic and M. Stojcev, "An overview of on-chip buses," *Facta Univ. - Ser. Electron. Energ.*, vol. 19, no. 3, pp. 405–428, 2006.

[35] I. O'Connor, F. Mieyeville, F. Gaffiot, A. Scandurra, and G. Nicolescu, "Reduction methods for adapting optical network on chip topologies to specific routing applications," *Proc. Des. Circuits Integr. Syst.*, no. October 2015, pp. 12–14, 2008.

[36] J. M. Senior and M. Y. Jamro, *Optical Fiber Communications: Principles and Practice*. 2009.

[37] M. . Flynn, Wayne, Luk, *Introduction to the Systems Approach*. 2008.

[38] S. W. Keckler, K. Olukotun, and H. P. Hofstee, *Multicore Processors and Systems*. 2009.

[39] X. Sun, R. Camacho-aguilera, C. Lionel, J. Michel, J. Liu, and L. C. Kimerling, "Ge-

on-Si laser operating at room temperature," *Opt. Lett.*, vol. 35, no. 5, pp. 679–681, 2010.

[40] J. Cardenas, C. B. Poitras, J. T. Robinson, K. Preston, L. Chen, and M. Lipson, "Low loss etchless silicon photonic waveguides," *Opt. Express*, vol. 17, no. 6, p. 4752, 2009.

[41] Z. Zhou, B. Yin, and J. Michel, "On-chip light sources for silicon photonics," *Light Sci. Appl.*, vol. 4, no. 11, p. e358, 2015.

[42] I. O'Connor, "Optical solutions for system-level interconnect," *Proc. 2004 Int. Work. Syst. Lev. interconnect Predict. - SLIP '04*, p. 79, 2004.

[43] S. Le Beux, G. Nicolescu, G. Bois, and P. Paulin, "A system-level exploration flow for Optical Network on Chip (ONoC) in 3D MPSoC," *ISCAS 2010 - 2010 IEEE Int. Symp. Circuits Syst. Nano-Bio Circuit Fabr. Syst.*, no. 1, pp. 3613–3616, 2010.

[44] J. E. Miller, H. Kasture, G. Kurian, C. Gruenwald Iii, N. Beckmann, C. Celio, J. Eastep, and A. Agarwal, "Graphite: A Distributed Parallel Simulator for Multicores."

[45] L. Bin and M. Samiei, "Comparison for Image Edge Detection Algorithms," *IOSR J. Comput. Eng.*, vol. 2, no. 6, pp. 1–4, 2012.

[46] I. Sobel, "An isotropic 3 by 3 image gradient operator," *Mach. Vis. three-demensional Sci.*, vol. 1, no. 1, pp. 23–34, 1990.

[47] P. P. Acharjya, R. Das, and D. Ghoshal, "A Study on Image Edge Detection Using the Gradients," vol. 2, no. 12, pp. 2–6, 2012.

[48] M. J. Flynn, "Some computer organizations and their effectiveness," *IEEE Trans. Comput.*, vol. C-21, no. 9, pp. 948–960, 1972.

[49] T. Lamine and B. Cheikh, "PARALLELIZATION STRATEGIES FOR MODERN

COMPUTING PLATFORMS: APPLICATION TO ILLUSTRATIVE IMAGE PROCESSING AND COMPUTER VISION APPLICATIONS."

[50] T. L. Ben Cheikh, G. Nicolescu, J. Trajkovic, Y. Bouchebaba, and P. Paulin, "Fast and accurate implementation of Canny edge detector on embedded many-core platform," in *2014 IEEE 12th International New Circuits and Systems Conference (NEWCAS)*, 2014, pp. 401–404.

[51] "Types of Bitmaps," 2012. [Online]. Available: http://msdn.microsoft.com/en-us/library/ms536393(v=vs.85).aspx. [Accessed: 05-Oct-2016].

[52] A. T. Tran, "On-Chip Network Designs for Many-Core Computational Platforms - Dissertation," 2009.

[53] C. Sun, C.-H. O. Chen, G. Kurian, L. Wei, J. Miller, A. Agarwal, L.-S. Peh, and V. Stojanovic, "DSENT - A Tool Connecting Emerging Photonics with Electronics for Opto-Electronic Networks-on-Chip Modeling," in *2012 IEEE/ACM Sixth International Symposium on Networks-on-Chip*, 2012, pp. 201–210.

[54] C. Sun, C. O. Chen, G. Kurian, J. Miller, A. Agarwal, L. Peh, and V. Stojanovic, "NoC Cost Evaluation is Critical," 2012.

[55] C. Dubnicki, "The Effects of Block Size on the Performance of Coherent Caches in Shared-Memory Multiprocessors," 1993.

Appendix

A1.1 Configurations for varying Image Sizes

Appendix Table 1: Configurations for varying Image Sizes

Configuration Parameters	Values
Synchronization Scheme	Lax Barrier
Image Size	512/1024/2048 pixels
Number of Cores	64
Number of Clusters	16
Routing Strategy	Cluster based
Flit Width	64 bits
Optical Access points per Cluster	4
Cache Line Size	32 bytes
L1 Data Cache Size	64 kB

A1.2 Configurations for varying Number of Cores

Appendix Table 2: Configurations for varying Number of Cores

Configuration Parameters	Values
Synchronization Scheme	Lax Barrier
Image Size	1024 pixels
Number of Cores	16/64/256
Number of Clusters	16
Routing Strategy	Cluster based
Flit Width	64 bits
Optical Access points per Cluster	4
Cache Line Size	32 bytes
L1 Data Cache Size	64 kB

A1.3 Configurations for varying Number of Clusters

Appendix Table 3: Configurations for varying Number of Clusters

Configuration Parameters	Values
Synchronization Scheme	Lax Barrier
Image Size	1024 pixels
Number of Cores	64
Number of Clusters	1/2/4/8
Routing Strategy	Cluster based
Flit Width	64 bits
Optical Access points per Cluster	4
Cache Line Size	32
L1 Data Cache Size	64

A1.4 Configurations for varying Number of Optical Access Points

Appendix Table 4: Configurations for varying Number of Optical Access Points

Configuration Parameters	Values
Synchronization Scheme	Lax Barrier
Image Size	1024 pixels
Number of Cores	64
Number of Clusters	8
Routing Strategy	Cluster based
Flit Width	64 bits
Optical Access points per Cluster	0/2/4/8
Cache Line Size	32
L1 Data Cache Size	64

A1.5 Configurations for varying Routing Strategies

Appendix Table 5: Configurations for varying Routing Strategies

Configuration Parameters	Values
Synchronization Scheme	Lax Barrier
Image Size	1024 pixels
Number of Cores	64
Number of Clusters	8
Routing Strategy	Cluster based/ Routing based
Flit Width	64 bits
Optical Access points per Cluster	4
Cache Line Size	32
L1 Data Cache Size	64

A1.6 Configurations for varying Flit Widths

Appendix Table 6: Configurations for varying Flit Width values

Configuration Parameters	Values
Synchronization Scheme	Lax Barrier
Image Size	1024 pixels
Number of Cores	64
Number of Clusters	8
Routing Strategy	Cluster based
Flit Width	16/32/64 bits
Optical Access points per Cluster	4
Cache Line Size	32
L1 Data Cache Size	64

A1.7 Configurations for varying Cache Line Sizes

Appendix Table 7: Configurations for varying Cache Line Sizes

Configuration Parameters	Values
Synchronization Scheme	Lax Barrier
Image Size	512 pixels
Number of Cores	64
Number of Clusters	4
Routing Strategy	Cluster based
Flit Width	64 bits
Optical Access points per Cluster	4
Cache Line Size	8/16/32/64
L1 Data Cache Size	64

A1.8 Configurations for varying L1 Data Cache Sizes

Appendix Table 8: Configurations for varying L1 Data Cache Sizes

Configuration Parameters	Values
Synchronization Scheme	Lax Barrier
Image Size	512 pixels
Number of Cores	64
Number of Clusters	4
Routing Strategy	Cluster based
Flit Width	64 bits
Optical Access points per Cluster	4
Cache Line Size	32
L1 Data Cache Size	8/16/32/64/128

A2.1 All Results for the Sequential Implementation Scheme

Appendix Table 9: Results for Sequential Implementation Scheme

Parameters Varied	Values	Latency (ns)			Contention Delay (ns)			Total Energy (J)		
		Emesh	ATAC	ORNoC	Emesh	ATAC	ORNoC	Emesh	ATAC	ORNoC
No. of Cores	16	12.712	13.112	13.112	0.235	0.135	0.135	0.032	0.041	0.04
	64	20.247	14.402	14.403	0.193	0.077	0.078	0.129	0.193	0.181
	256	37.187	15.782	15.783	0.3	0.139	0.14	0.527	0.587	0.586
Image Sizes	512	20.182	14.416	14.417	0.136	0.099	0.1	0.032	0.043	0.043
	1024	20.247	14.448	14.442	0.193	0.101	0.102	0.129	0.181	0.176
	2048	20.285	14.578	14.573	0.243	0.112	0.11	0.527	0.707	0.697
No. of Clusters	1	20.182			0.136			0.032		
	2		17.565	17.566		0.081	0.085		0.035	0.035
	4		15.811	15.812		0.106	0.108		0.039	0.038
	8		14.418	14.42		0.099	0.102		0.049	0.046
Flit Width	16	35.659	31.291	31.292	4.114	4.088	4.088	0.041	0.055	0.045
	32	25.394	20.796	20.795	1.511	1.254	1.251	0.070	0.097	0.079
	64	20.247	15.796	15.798	0.193	0.084	0.085	0.129	0.190	0.147
No. of OAPC	0	20.245			0.192			0.002		
	2		17.337	17.334		0.11	0.11		0.001	0.0009
	4		15.797	15.797		0.086	0.086		0.001	0.0009
	8		14.279	14.277		0.079	0.079		0.001	0.0009
Routing Strategy	XY	20.247			0.193			0.129		
	distance		14.161	14.165		0.073	0.078		0.193	0.181
	cluster		14.403	14.405		0.078	0.081		0.193	0.181
Cache Line Size	8	18.004	15.483	15.482	0.074	0.117	0.116	0.040	0.044	0.043
	16	18.718	16.158	16.16	0.084	0.086	0.088	0.036	0.038	0.038
	32	20.182	17.563	17.567	0.135	0.081	0.085	0.032	0.036	0.035
	64	23.716	20.96	20.968	0.853	0.668	0.67	0.031	0.034	0.034
Cache Size	8	20.247	17.59	17.591	0.192	0.098	0.1	0.13	0.143	0.141
	16	20.246	17.589	17.592	0.192	0.099	0.102	0.13	0.142	0.14
	32	20.247	17.588	17.591	0.193	0.099	0.102	0.129	0.141	0.14
	64	20.236	17.579	17.581	0.192	0.099	0.102	0.129	0.141	0.139
	128	20.235	17.577	17.58	0.193	0.098	0.102	0.129	0.141	0.139

A2.2 All Results for the Pure data Implementation Scheme

Appendix Table 10: Results for Pure Data Implementation Scheme

Parameters Varied	Values	Latency(ns)			Contention Delay(ns)			Total Energy(J)		
		Emesh	ATAC	ORNoC	Emesh	ATAC	ORNoC	Emesh	ATAC	ORNoC
No. of Cores	16	12.971	13.109	13.105	0.202	0.117	0.117	0.039	0.049	0.048
	64	20.1	13.961	13.962	0.16	0.062	0.062	0.153	0.232	0.218
	256	36.761	15.179	15.173	0.24	0.128	0.127	0.63	0.703	0.703
Image Sizes	512	20.025	13.812	13.813	0.116	0.071	0.073	0.04	0.06	0.059
	1024	20.1	13.953	13.956	0.16	0.082	0.08	0.154	0.227	0.223
	2048	20.23	14.21	14.202	0.187	0.093	0.091	0.619	0.858	0.845
No. of Clusters	1	20.025			0.116			0.04		
	2		17.209	17.215		0.07	0.071		0.045	0.045
	4		15.431	15.435		0.086	0.088		0.049	0.048
	8		13.953	13.956		0.082	0.085		0.062	0.058
Flit Width	16	36.356	31.77	31.769	3.3	3.285	3.282	0.053	0.072	0.059
	32	25.456	20.67	20.676	1.209	0.995	0.999	0.092	0.137	0.104
	64	20.098	15.357	15.357	0.159	0.086	0.086	0.154	0.231	0.175
No. of OAPC	0	20.011			0.171			0.17		
	2		16.896	16.895		0.099	0.097		0.191	0.187
	4		15.353	15.357		0.078	0.076		0.195	0.19
	8		14.776	14.777		0.067	0.066		0.203	0.197
Routing Strategy	XY	20.1			0.16			0.154		
	distance		13.854	13.853		0.056	0.057		0.232	0.218
	cluster		13.96	13.968		0.061	0.062		0.232	0.218
Cache Line Size	8	17.394	14.673	14.676	0.054	0.099	0.099	0.047	0.052	0.051
	16	18.284	15.524	15.529	0.073	0.079	0.083	0.043	0.047	0.047
	32	20.024	17.216	17.22	0.117	0.07	0.072	0.040	0.045	0.045
	64	23.934	21.011	21.052	0.677	0.54	0.546	0.039	0.043	0.043
Cache Size	8	20.099	17.245	17.253	0.159	0.079	0.081	0.155	0.170	0.168
	16	20.103	17.249	17.252	0.162	0.08	0.082	0.154	0.170	0.168
	32	20.102	17.253	17.262	0.16	0.082	0.083	0.154	0.169	0.168
	64	20.103	17.252	17.254	0.161	0.081	0.081	0.154	0.169	0.167
	128	20.065	17.22	17.219	0.162	0.081	0.082	0.153	0.169	0.166

A2.3 All Results for the Mixed Implementation Scheme

Appendix Table 11: Results for Mixed Parallelism Implementation Scheme

Parameters Varied	Values	Latency(ns)			Contention Delay(ns)			Total Energy(J)		
		Emesh	ATAC	ORNoC	Emesh	ATAC	ORNoC	Emesh	ATAC	ORNoC
No. of Cores	16	12.867	13.055	13.052	0.18	0.114	0.113	0.038	0.049	0.048
	64	20.037	13.925	13.924	0.152	0.059	0.058	0.152	0.23	0.216
	256	36.715	14.992	15.141	0.233	0.122	0.121	0.624	0.696	0.696
Image Sizes	512	19.902	13.851	13.852	0.101	0.068	0.069	0.04	0.055	0.054
	1024	20.037	13.947	13.944	0.152	0.078	0.08	0.152	0.225	0.219
	2048	20.18	14.197	14.195	0.18	0.087	0.086	0.614	0.83	0.82
No. of Clusters	1	19.902			0.101			0.04		
	2		17.108	17.113		0.068	0.072		0.044	0.044
	4		15.328	15.328		0.082	0.084		0.048	0.048
	8		13.853	13.863		0.076	0.076		0.061	0.057
Flit Width	16	36.655	32.084	32.095	3.788	3.772	3.777	0.048	0.065	0.053
	32	25.453	20.726	20.731	1.244	1.07	1.07	0.082	0.118	0.093
	64	20.034	15.401	15.401	0.152	0.068	0.068	0.152	0.224	0.172
No. of OAPC	0	20.034			0.152			0.153		
	2		16.941	16.94		0.086	0.085		0.172	0.168
	4		15.403	15.398		0.069	0.067		0.175	0.171
	8		14.694	14.694		0.056	0.055		0.183	0.178
Routing Strategy	XY	20.037			0.152			0.152		
	distance		13.926	13.81		0.059	0.063		0.233	0.216
	cluster		13.927	13.926		0.06	0.07		0.233	0.217
Cache Line Size	8	17.422	14.713	14.714	0.047	0.089	0.089	0.048	0.053	0.052
	16	18.259	15.517	15.515	0.066	0.077	0.074	0.044	0.047	0.047
	32	19.902	17.113	17.116	0.101	0.067	0.073	0.040	0.044	0.044
	64	23.626	20.74	20.721	0.649	0.517	0.508	0.039	0.043	0.043
Cache Size	8	20.035	17.217	17.212	0.151	0.084	0.084	0.154	0.168	0.166
	16	20.036	17.214	17.213	0.152	0.082	0.084	0.153	0.167	0.164
	32	20.037	17.214	17.211	0.152	0.081	0.084	0.153	0.167	0.164
	64	20.032	17.203	17.207	0.152	0.079	0.082	0.153	0.166	0.163
	128	20.006	17.169	17.178	0.151	0.08	0.084	0.152	0.165	0.162