GPUOPT: Power Efficient Photonic Network-on-Chip for a Scalable GPU

JANIBUL BASHIR, Department of Computer Science and Engineering, Indian Institute of Technology, New Delhi, India
SMRUTI R. SARANGI, Department of Computer Science and Engineering (joint appointment with Department of Electrical Engineering), Indian Institute of Technology, New Delhi, India

On-chip photonics is a disruptive technology, and such NoCs are superior to traditional electrical NoCs in terms of latency, power, and bandwidth. Hence, researchers have proposed a wide variety of optical networks for multicore processors. The high bandwidth and low latency features of photonic NoCs have led to the overall improvement in the system performance. However, there are very few proposals that discuss the usage of optical interconnects in Graphics Processor Units (GPUs). GPUs can also substantially gain from such novel technologies because they need to provide significant computational throughput without further stressing their power budgets.

The main shortcoming of optical networks is their high static power usage because the lasers are turned on all the time by default, even when there is no traffic inside the chip, and thus sophisticated laser modulation schemes are required. Such modulation schemes base their decisions on an accurate prediction of network traffic in the future. In this paper, we propose an energy-efficient and scalable optical interconnect for modern GPUs called *GPUOPT* that smartly creates an overlay network by dividing the symmetric multiprocessors (SMs) into clusters. It furthermore has separate sub-networks for coherence and non-coherence traffic. To further increase the throughput we connect the off-chip memory with optical links as well.

Subsequently, we show that traditional laser modulation schemes (for reducing static power consumption) that were designed for multicore processors are not that effective for GPUs. Hence, there was a need to create a bespoke scheme for predicting the laser power usage in GPUs.

Using these set of techniques, we were able to improve the performance of a modern GPU by 45% as compared to a state-of-the-art electrical NoC. Moreover, as compared to competing optical NoCs for GPUs, our scheme reduces the laser power consumption by 67%, resulting in a net 65% reduction in ED^2 for a suite of Rodinia benchmarks.

CCS Concepts: • **Networks → Photonic Network on chip**; • **Computer systems organization → Optical Interconnection architectures**; **System on a chip**.

Additional Key Words and Phrases: on-chip networks, photonics, static power consumption, GPUs

ACM Reference Format:
Janibul Bashir and Smruti R. Sarangi. 2020. GPUOPT: Power Efficient Photonic Network-on-Chip for a Scalable GPU. *ACM J. Emerg. Technol. Comput. Syst.* 1, 1, Article 1 (January 2020), 25 pages. https://doi.org/10.1145/3416850

Authors' addresses: Janibul Bashir, Department of Computer Science and Engineering, Indian Institute of Technology, New Delhi, India, janibbashir@cse.iitd.ac.in; Smruti R. Sarangi, Department of Computer Science and Engineering (joint appointment with Department of Electrical Engineering), Indian Institute of Technology, New Delhi, India, srsarangi@cse.iitd.ac.in.

1550-4832/2020/1-ART1 $15.00
https://doi.org/10.1145/3416850

1 INTRODUCTION

GPUs and GPGPUs (General Purpose Graphics Processing Units) are now regarded as first-class processing devices when we need to run numerical and HPC workloads [54, 56] [1]. Their complexity has increased over the years, and the latest state-of-the-art GPUs have thousands of cores, and a hierarchy of memory structures within them.

This massive integration of processing elements in GPUs enables such architectures to execute thousands of threads simultaneously [45, 47]. As a result of the sizeable computational load, the processing elements generate a huge amount of traffic in order to communicate with each other, as well as with the off-chip memory modules [46]. To handle this traffic, researchers initially moved from on-chip electrical buses to electrical on-chip networks (ENoCs) [16, 64], which resulted in considerable improvements in terms of latency and instruction throughput. However, with an increase in the number of on-chip elements in modern GPUs the performance of ENoCs is showing signs of saturation [22, 25, 66]. For further scalability we need to opt for new network technologies.

Silicon photonics[2] [5, 7, 30, 52] with 2.5D integration [26, 58] (two layers: logical layer and photonics layer) is a *worthy* competitor in this space. Photonics has several advantages: extremely low latency (2-3 cycles from any point to any other point), transmission power that is independent of the distance, and high bandwidth.

On-chip photonics is showing signs of maturity. As of now on-chip photonics is mainly being used to create optical routers in fast communication technologies such as 100G Ethernet. The next step is to integrate these photonic components with a logic layer. In this space numerous working prototypes have been created by researchers in academia and industry [37, 44]. Numerous fabrication houses [52] have started fabricating small chips that have a photonics layer and logic layer. In many such chips, the low latency of transmission and the possibility of high-bandwidth transmission because of wavelength division multiplexing has been demonstrated [13, 51].

1.1 Challenges

Photonic NoCs (PNoCs) proposed for multi-core CPUs cannot be directly used in GPUs. For example, in CPUs, coherence messages (multicast and broadcast messages including cache coherence and barrier messages) form a significant component of on-chip traffic [4, 6], however, in GPUs coherence traffic contributes to less than 2% of the overall on-chip traffic (see Section 4.3.3). The design of the PNoC needs to be modified to take this in account. GPUs have far more connected components, and the access pattern is also more regular, whereas CPUs have a very different on-chip cache access pattern. All of these reasons make it necessary to create a bespoke PNoC for a GPU.

The state-of-the-art optical NoC for GPUs uses different types of optical crossbars [65]. Along with several performance issues the major shortcoming of this work is that it has not addressed the issue of high static power consumption [4, 5]. In PNoCs, we have to continuously pump in light into the chip even when there is reduced traffic inside the chip, thus resulting in a huge amount of static power consumption. The reason stems from the inherent property of light – we cannot store photons.

The issue of high static power consumption has been extensively addressed in PNoCs for CPUs. The standard approach is to divide the total execution time into fixed-size *epochs*. We predict the laser power usage of the next epoch in advance based on network usage, and then modulate the laser power accordingly [6, 50, 62]. However, for GPUs, no such scheme has been proposed till now. Thus, it is necessary to develop a GPU-specific scheme.

[1]This paper is an extension of a 2-page paper published in NOCS 2019 [8]
[2]We shall use the terms *optical* and *photonics* interchangeably

1.2 Our Solution

We characterize the behavior of modern GPU workloads and based on the characterization results, we derive some GPU-specific insights. Based on those insights, we propose to combine two emerging technologies (2.5D integration and photonics) to design a novel GPU-specific scalable PNoC, *GPUOPT*. We propose a 2.5D chip in which we have a logical layer (containing processor cores and cache banks) stacked on top of the photonic layer. The photonic layer includes the photonic components required for on-chip communication. For an optical network, we intelligently divide the logical components into different types of clusters in order to decrease the contention inside the network. Moreover, to further improve the performance we propose to route unicast and multicast messages separately via different photonic sub-networks. Additionally, we tried to address the primary concern of PNoCs – high static power consumption – by proposing a GPU specific prediction scheme for modulating the off-chip light source. This reduced the laser power usage significantly ($\approx 67\%$).

1.3 Summary of Results

To evaluate the potential benefits of our proposed scheme, we evaluated a GPU with 64 SMs (streaming multiprocessors). We used workloads from the *Rodinia* benchmark suite [12] and compared our scheme with a high-performing electrical on-chip network (*EMESH*) [64] and a state-of-the-art PNoC (*Prior_Opt*) [65]. The results show that our scheme performs (reciprocal of simulated execution time) 45% and 17% better than *EMESH* and *Prior_Opt* respectively. Moreover, using the laser modulation scheme, our design resulted in a 67% decrease in static power consumption, thereby reducing the ED^2 (energy delay square) by 65% as compared to *Prior_Opt*.

1.4 Contributions

The main contributions of this paper are:

❶ We analyzed the behavior of modern GPU workloads and explored the benefits of using a high bandwidth and low latency on-chip network in futuristic GPUs.

❷ We designed a 2.5D-stacked photonics-enabled GPU chip for 2048 in-order processor cores (with 64 SMs).

❸ We proposed to separate the on-chip communicating components into different types of clusters in order to enhance the overall performance of the system.

❹ We proposed to route the coherence traffic through a single-writer-multiple-reader based PNoC and used a multiple-reader-single-writer based PNoC for non-coherence traffic.

❺ We modulated the off-chip light source by designing a novel prediction scheme based on past network behavior in order to reduce static power consumption.

❻ We evaluated our design by comparing it with an electrical on-chip network and a state-of-the-art PNoC for a suite of *Rodinia* benchmarks.

1.5 Organization of the Paper

The rest of the paper is organized as follows. Section 2 shows the organization of the GPU and its execution model. Background concepts related to photonics are discussed in Section 3. Section 4 characterizes the behavior of modern GPU workloads and derives various insights, and based on those insights we discuss the architecture of our proposed photonic NoC in Section 5. The laser modulation technique for reducing static power consumption is then discussed in Section 6. The simulation methodology is described in Section 7. In Section 8, we evaluate our design, discuss related work in Section 9, and finally conclude in Section 10.

2 GPU ORGANIZATION AND DESIGN

In this section, we shall describe the design of modern GPGPUs [34, 45, 47]. For the sake of simplicity we shall use the term *GPU* to refer to the general purpose part of a GPGPU. Furthermore, we shall broadly follow NVIDIA's Pascal architecture and associated terminology [15, 55] in the rest of this paper.

2.1 Architecture

In almost all GPUs, three fundamental components shape the overall architecture: thread engine (command processor), streaming multiprocessor (compute unit) and the memory hierarchy [45, 47]. Let us briefly discuss each component.

Thread Engine: The basic functionality of a GPU is to execute thousands of threads simultaneously on different cores. A *host* program on the CPU offloads thousands of such threads (*kernel*) to the GPU. Inside the GPU the thread engine schedules blocks of such threads on different execution engines known as streaming multiprocessors or SMs.

Streaming Multiprocessor (SM): An SM is the primary execution engine inside a GPU. A single GPU contains tens of SMs with each SM containing multiple streaming processors (SP), load store units (LSU) and special functional units (SFU). SPs inside an SM work in single-instruction multiple-data (SIMD) fashion (executing the same instruction on different data) and thus making a GPU highly parallel execution framework. The load and store operations inside an SM are performed using LSUs. SFUs are required to execute transcendental instructions, and also offload some of the numerical computation. Figure 1 (a) shows the high level architectural view of a GPU. In Figure 1(b) we show the organization of different components inside each SM.

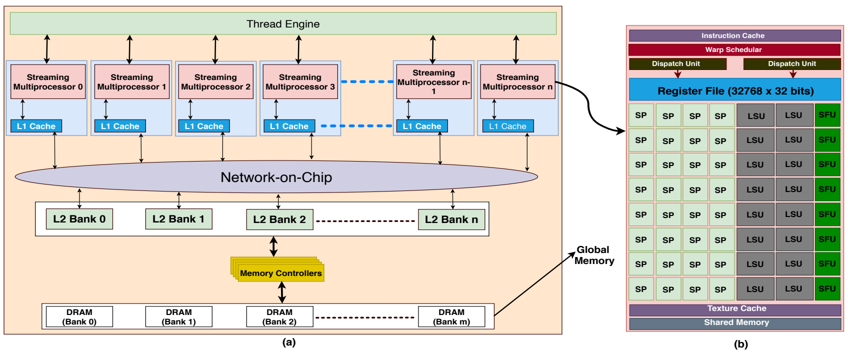

Fig. 1. (a) High level architectural diagram of a representative GPU (b) Organization of an SM containing 32 SPs, 16 LSUs and 8 SFUs. Adapted from [63]

GPU Memory Hierarchy: In almost all NVIDIA GPUs, there are generally six different types of memories: register memory, constant memory, shared memory, local memory, texture memory, and global memory. Here we will briefly discuss the global memory as it is the only memory relevant to the understanding of our work. For further details interested readers may refer to [15, 55].

All the threads inside a *kernel* (function to be executed on a GPU) cooperate through global memory. In almost all modern GPUs, the global memory accesses are cached at various levels inside the chip. As a result, contemporary GPUs have different levels of caches [15, 55]. Figure 1 shows two levels: L1 and L2 [46]. The L1 cache is located inside each SM and is private to the SM, whereas the L2 cache is accessible by all the threads inside the *kernel.* The L2 cache in modern NVIDIA GPUs is physically partitioned into several banks and each bank is coupled with a separate memory controller (MC) [46].

2.2 Execution Model of a GPU

In GPU terminology, a *kernel* is a function that is to be executed by a GPU. Each *kernel* is executed as a grid of thread blocks. The *host* offloads the grid to the GPU thread engine. The thread engine schedules the thread blocks on different SMs inside a GPU. Internally, each SM executes threads in groups called *warps*. The threads inside a *warp* execute simultaneously on different SPs. The warp scheduler inside an SM performs the function of scheduling the warps inside each block. Each thread inside a block has its own set of registers and local memory. In order to cooperate with other threads inside its block, it uses the shared memory region located inside its SM. However, threads belonging to different thread blocks coordinate only through global memory. In a nutshell, we have an SP executing a single thread, then an SM executing a block of threads and an entire GPU device executing a grid of such blocks.

3 BACKGROUND OF PHOTONICS

3.1 An Optical Network

An optical network has four basic components: laser light source – to generate optical power, modulator – to encode data by modulating the optical signal, waveguide – to carry optical signals, and a photodetector – to detect the value of the optical signal. Let us briefly describe each component.

Light Source: Lasers are the most commonly used light source in photonic on-chip networks [4–7, 60]. In our proposed PNoC, we use an array of 32 directly modulated (DML) off-chip lasers as the light source. These lasers are commercially available [9, 18–20], can be modulated at GHz frequency and are thermally stable. The main aim behind using an array of such lasers is that we can increase or decrease the optical power by turning some lasers on and off in the array. On-chip light sources are also available [10, 36]. However, they consume a lot of on-chip power and area and dissipate the entire heat inside the chip [40]. Hence, we only use off-chip lasers.

Waveguide: Like electrical wires in ENoCs, we have silicon waveguides [11] in photonic on-chip networks. The light is guided and confined within the waveguide by using a combination of a low refractive index material as a cladding (outside) and a high refractive index material as the core of the waveguide (inside). In our design, we assume silicon waveguides with a propagation loss less than 0.5dB/cm [11]; they are capable of carrying 64 wavelengths [11] together using DWDM (dense wavelength division multiplexing).

Modulator: A modulator is an optical device that is used to turn an optical signal on or off. In a dense wavelength division multiplexing (DWDM) based optical network, multiple wavelengths traverse through the same link. Thus, we require wavelength-selective modulators in such networks so that each modulator modulates its corresponding wavelength accordingly. Such modulators have already been proposed in prior work with a modulation rate higher than 10Gb/s [53] and we have assumed the same modulators in our design. These are silicon ring resonator [53] based modulators. A ring resonator is an optical device, which can be used to inject or filter out a specific wavelength, called the *resonant wavelength*, from the optical waveguide.

Photodetector: At the receiver side, we require an optical filter and a photodetector. A ring resonator based filter extracts a specific wavelength from the waveguide and guides this light to the photodetector. The photodetector extracts the incident optical signal and converts it to an electrical signal. Both these objectives can be achieved using a Germanium doped ring resonator [41, 60]. We have used one such wavelength selective detector [60] in our design. This detector has a capacitance less than 1fF and occupies an area equal to $20\mu m^2$.

In addition to these four components, we use two additional components: an optical power splitter and a comb splitter. Optical power splitters are required to split the laser-generated monochromatic light (at 1550 nm) between the different power waveguides [23]. The amount of light split depends on the split ratio of the optical power splitter. The splitter, whose split ratio can be tuned dynamically, is called a *tunable optical power splitter* [23]. In our design, we use a tunable power splitter that can be tuned in a single cycle (see [23] for the specifications).

Subsequently, comb splitters [32] are required to create multiple carrier signals at different wavelengths. We have such a splitter at each *optical station* (transmitter + receiver) to convert the monochromatic light into signals at different equispaced wavelengths. Each such wavelength can then be separately modulated to carry information (E/O ($electrical \rightarrow optical$) conversion).

3.2 A Note about Feasibility

All the components that we have used in our design are either commercially available (such as the off-chip DML lasers), are being used in other commercial devices such as fast ethernet switches (tapers, splitters and ring resonators), or very mature prototypes have been created in academia and industry (such as 2.5D integration).

3.3 Optical Links

An optical link is a group of waveguides used as a communication medium between two different communicating stations. Based on the number of stations reading and writing data from a link, optical links are divided into the following three categories:

(1) **Single Writer Multiple Reader (SWMR):** In an SWMR link [49, 50] (or waveguide), only one specific station is allowed to write its data, whereas all other stations can read data from the link. All the receivers attached to a waveguide are by default turned off. When a station writes its data on the waveguide, it informs the intended receiver to turn on its detectors for incoming data [49] using a separate *reservation waveguide*.
(2) **Multiple Writer Single Reader (MWSR):** In an MWSR link [60] every station is allowed to write its data on the shared link. However, only one specific station is permitted to read data from the link. An arbitration scheme is required to provide mutual exclusion at the senders' side [59].
(3) **Multiple Writer Multiple Reader (MWMR):** An MWMR link [4, 48] is a hybrid of SWMR and MWSR. Multiple stations are allowed to read and write data from the same link. For such links, we require arbitration at the senders' side and reservation at the receivers' side.

3.4 Crosstalk Noise

In any DWDM (dense wavelength division multiplexing) based optical interconnect, different wavelengths are traversing through a same waveguide. In addition, there are numerous wavelength selective optical components such a modulators, filters, and splitters used in such networks. As a result, the different wavelengths interfere with each other through different optical components, thereby generating crosstalk noise [17, 61]. The generated crosstalk noise not only degrades the system performance but also affects the scalability of the overall system. Our topology is designed in such a way in order to decrease the generated crosstalk noise. We propose to use a serpentine shaped topology where different waveguides run parallel to each other, thereby eliminating the crosstalk due to waveguide crossings. Moreover, it reduces the crosstalk because of lesser waveguide bends in such a topology. To calculate the final crosstalk noise and subsequently the minimum energy per bit required to reach the farthest station, we use the analytical model proposed by Duong et al. [17].

4 BENCHMARK CHARACTERIZATION AND MOTIVATION

The aim of this section is to run GPU workloads on a GPU with a state of the art electrical NoC, and then characterize the benchmarks.

4.1 Baseline Architecture

Our baseline GPU architecture is tile-based. There are two different types of tiles in our design: *SM_Tile* and *LLC_Tile*. Each *SM_Tile* is composed of 8 SMs with each SM containing 32 SPs, 16 LSUs, and 8 SFUs as shown in Figure 1 (b). An *LLC_Tile* has one L2 bank and one memory controller. These tiles are placed on the GPU chip in a grid. Intra-tile communication is done using point-to-point electrical links, whereas, for inter-tile communication each tile has an on-chip electrical router [28]. These routers are connected using an NoC. In our configuration, we have assumed a 4×4 electrical mesh [28, 64] based NoC. This simple design performed the best in our simulations (for detailed architectural parameters, please refer to Table 2).

4.2 GPU Workloads

We used 10 CUDA applications from the *Rodinia* benchmark suite (rest were not compatible with CUDA version 6.0) [12], and simulated them using the cycle-accurate architectural simulator, *GPUTejas* (validated with native hardware) [38]. Note that we define **performance** as the reciprocal of the simulated execution time.

4.3 Characterization

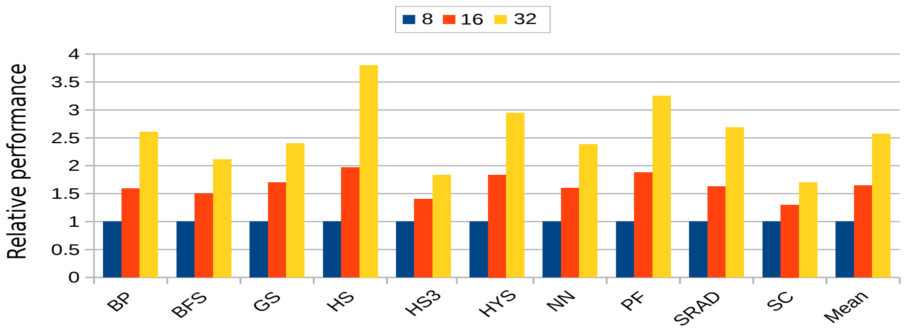

Fig. 2. Relative performance comparison across different number of SMs

4.3.1 Scalability. Figure 2 shows the effect of the number of SMs (8, 16 or 32) on the overall performance of the system. It is clear from the graph that with an increase in the number of SMs in a GPU, there is a significant change (up to 3.8X) in the performance of the system. In benchmarks such as *HS*, *HYS*, and *PF* the improved performance (2.9-3.8X) is attributed to a higher degree of parallelism in such benchmarks. However, in the case of benchmarks such as *BFS* and *SC*, the low performance scalability is due to greater NoC activity (see Section 4.3.2). On an average, there is a 2.6X improvement in performance when we move from a 8 to 32-SM configuration. Summary: *For many workloads we can achieve further scalability by increasing the number of SMs; however, for many others we need to increase the performance of the NoC such that memory accesses can be serviced sooner.*

4.3.2 Breakup of Memory Accesses. In Figure 3 we have clubbed all the memory accesses that use the NoC into the same category (such as L2/memory), and then shown the relative fraction of accesses in each category. We observe from the figure that in benchmarks such as *BFS*, *HS3*, *NN*,

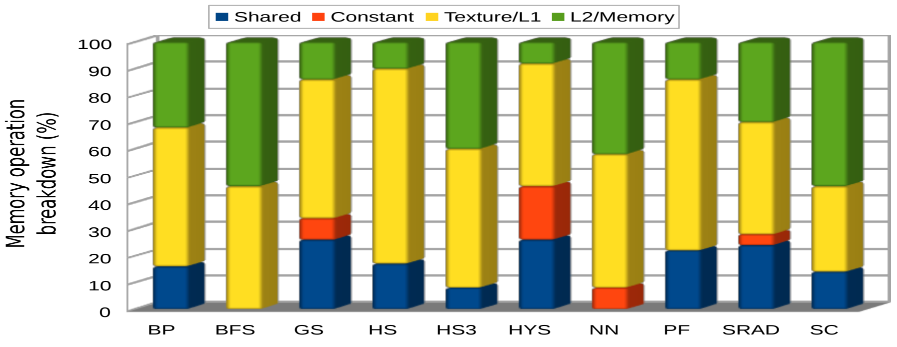

Fig. 3. Memory access breakup

and *SC*, the number of L2/memory accesses is more than 40%. This results in higher NoC activity in such benchmarks, which in turn leads to the limited performance scalability of such benchmarks (correlated with Figure 2). This motivates the need for fast and high-bandwidth interconnects such as optical NoCs (**Insight:Optical**).

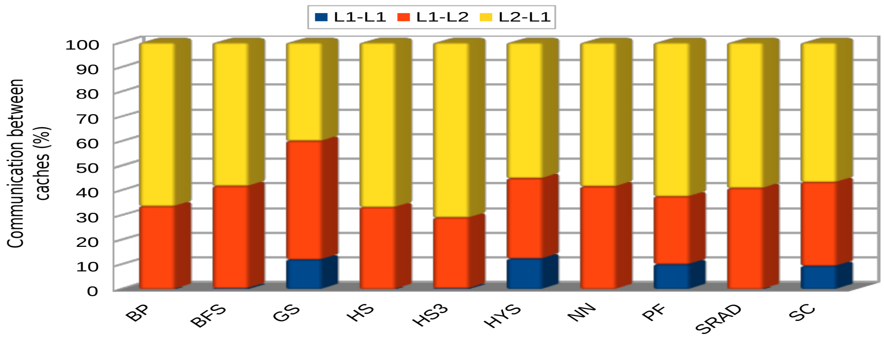

Fig. 4. Breakup of the communication between the L1 and L2 caches

4.3.3 Characterization of the Communication between Caches. Figure 4 shows the breakup of communication between the different types of caches (L1 and L2). We observe from the figure that there is a negligible amount of communication between the L1 caches. L1 to L1 communication is mainly because of false sharing and the limited amount of coherence traffic– these are either multicast or broadcast messages. Hence, a single sender multiple receiver-based topology should be used for such messages (**Insight:SWMR**). Additionally, the lower L1 to L2 communication is attributed to different architectural enhancements such as memory coalescing [21]. Most of the communication ($\approx 60\%$) is dominated by $L2 \longrightarrow L1$ communication, which is mostly reading data and filling the caches. Thus, the traffic from the L2 units dominate the NoC traffic and hence they require much more bandwidth as compared to the L1 units. Hence, we arrive at **Insight:LLC-Cluster** – each L2 unit should be associated with a dedicated communication interface to provide better access to the NoC. Also, because of lesser L1 to L2 communication, we can have a single communication interface for multiple L1 units (SMs) – **Insight:SM-Cluster**.

We did further experiments to assess the sensitivity of the GPU performance with respect to the $L1 \longrightarrow L2$ and $L2 \longrightarrow L1$ delay. Our results indicate that the GPU performance is insensitive to the $L1 \longrightarrow L2$ communication delay (less than 0.23% decrease in performance for a 20 cycle increase in delay). In comparison, the $L2 \longrightarrow L1$ delay is slightly more sensitive (3.32% decrease in performance).

This is because the former type of traffic consists mostly of writes, and the latter consists mostly of reads that are often on the critical path.

Nevertheless, the overarching conclusion is that in both the cases GPU benchmarks' sensitivity to the NoC latency is *low*. This should not lead to the false conclusion that latency is unimportant. It manifests itself via the well known law in queuing theory called *Little's law* [35] that states that in simplistic settings the size of the NoC's buffers is roughly proportional to the bandwidth multiplied with the latency. If we set the buffer size as a constraint then to sustain high bandwidth, we need a low latency. In this case latency seems to be relatively unimportant because of our choice of buffer sizes.

4.3.4 Effect of the NoC Bandwidth. In order to compare the effect of the bandwidth on the overall performance of the GPU, we compared the performance of a GPU by varying the bandwidth of the underlying NoC. This was done by increasing the number of electrical links connecting the on-chip routers (hypothetically), thus transferring higher number of bytes per unit time. The comparison results are shown in Figure 5. We observe that for memory intensive benchmarks such as *BFS*, *NN*, and *SC*, the higher-bandwidth configuration shows a greater improvement in performance as compared to the lower-bandwidth configuration by up to 70%. For most benchmarks the additional speedup is in the range of 30-50%, which is significant. We thus arrive at **Insight:Optical** – use a high bandwidth interconnect.

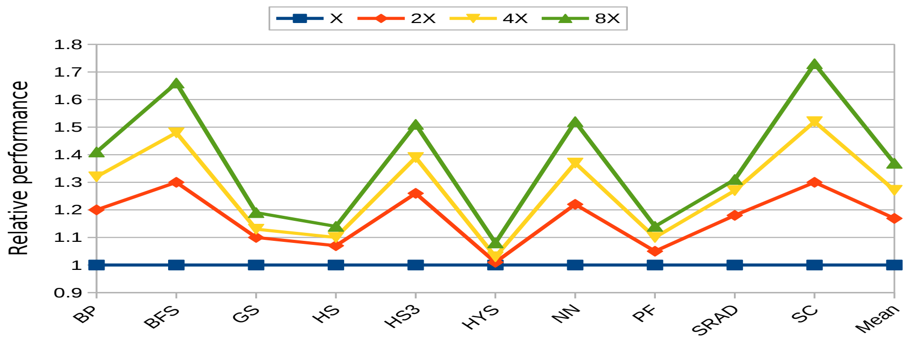

Fig. 5. GPU performance sensitivity to NoC bandwidth (X=0.5Tb/s)

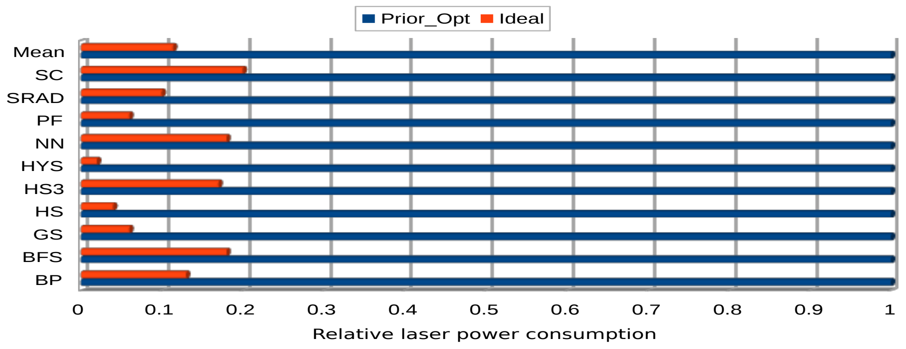

Fig. 6. Laser power consumption of the design in [65] as compared to an ideal system

4.3.5 Static Optical Power Consumption. One main disadvantage of PNoCs is the high laser power consumption. To understand the levels of power consumption, we compared the laser power consumption of an *ideal* scheme with a scheme that does not use any laser modulation technique.

For the ideal scheme, we assume one laser per optical station. The laser is by default turned off. It is turned on instantaneously only when an on-chip optical station wants to send a message, and for a scheme that does not use any laser modulation, we use the GPU PNoC proposed by Ziabari et al. [65] (Prior_Opt). It uses an SWMR bus for L2 to L1 communication and an MWSR bus for L1 to L2 communication. Figure 6 shows the comparison.

We observe that the Prior_Opt consumes 8.8X more laser power as compared to an ideal scheme. Thus, in order to reduce the static power consumption, it is necessary to modulate the off-chip laser based on predicting network activity (**Insight:Predict**). Additionally, the GPU performance is very insensitive to the latency of the L1 to L2 link, we can delay the messages originating from L1 units (SMs) (**Insight:Rigid**). However, the messages generated by L2 units should be sent as soon as possible (**Insight:Flexible**).

5 GPUOPT TOPOLOGY

In this section we propose a design of a PNoC, *GPUOPT*, based on the insights derived in Section 4.

5.1 Overview

Our topology aims at ❶ increasing the overall performance of the system by (a) decreasing contention in the network, and (b) providing a high bandwidth and low latency communication framework, and ❷ reducing the high static power consumption by (a) allowing stations to share the available on-chip power, and (b) tuning the off-chip laser power based on a predictive approach. To achieve these goals, we propose an efficient optical NoC called *GPUOPT*. Moreover, to reduce static power consumption, we introduce a hybrid laser modulation technique (described in Section 6). Finally, we incorporate our laser modulation technique in *GPUOPT* and develop an efficient optical NoC for GPUs. The NoC with the laser modulation scheme is referred to as *PS_GPUOPT*.

This section describes the topology of our proposed PNoC and in Section 6 we shall discuss our proposed GPU specific laser modulation technique.

5.2 Design

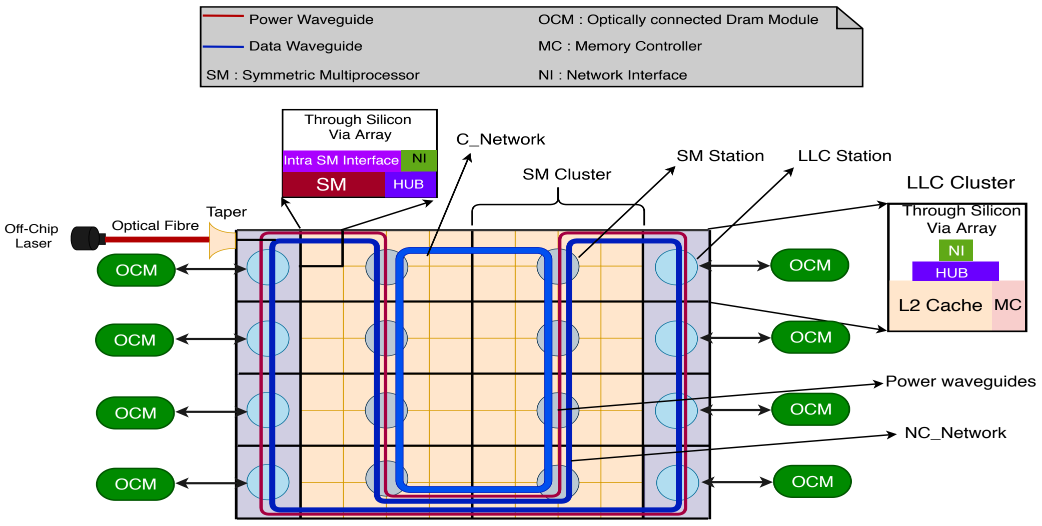

Fig. 7. Topology of an optical NoC for GPU

Figure 7 illustrates our proposed optical NoC (uses 2.5D integration with two stacked layers – optical and silicon). The figure shows the optical NoC, optically connected global memory modules, and other optical components. The chip has two separate layers: logical layer – containing SMs, L2 banks (last level cache), and memory controllers (MC), and the photonic layer containing optical components. The logical layer is divided into 16 clusters: 8 *SM_Clusters* and 8 *LLC_Clusters* [3]. Each *SM_Cluster* has 8 SMs where each SM contains a private L1 instruction and data cache.

Each *LLC_Cluster* has an L2 bank along with a memory controller. The intra-cluster communication is done electrically, whereas, for inter-cluster communication, we incorporate a separate silicon photonics layer underneath the logical layer. The optical layer has optical stations – one for each cluster, and these stations are connected using an optical crossbar (see Section 5.3). The clusters are connected to their respective optical stations using through-silicon vias (TSVs). Note that the optical stations attached to *SM_clusters* are called *SM_stations*, whereas the stations connected to *LLC_clusters* are called *LLC_stations*.

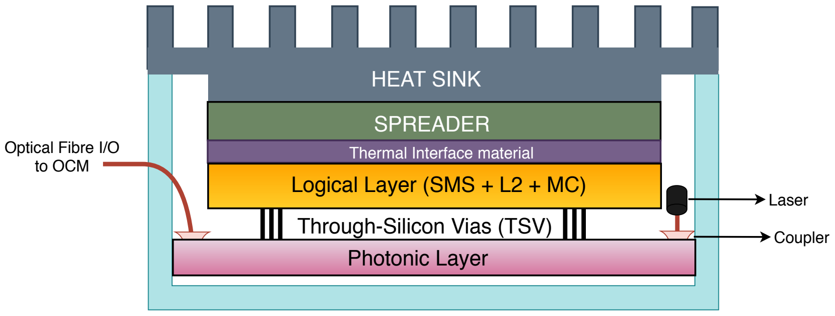
HEAT SINK
SPREADER
Thermal Interface material
Logical Layer (SMS + L2 + MC)
Through-Silicon Vias (TSV)
Photonic Layer
Optical Fibre I/O to OCM
Laser
Coupler

Fig. 8. Cross-sectional view of the layout

Figure 8 shows the cross-sectional view of our chip containing different layers. Most of the activity and heat generation happens in the silicon die and thus it is placed directly beneath the heat sink. The main reason behind choosing 2.5D integration is the better area utilization and easy integration of different technology layers.

5.3 Optical Crossbar

5.3.1 Data Network. In a photonic network-on-chip (PNoC), a data network is required to route the data-carrying optical signals from a source to a destination. In our topology, we use two different optical data networks – The *C_network* and *NC_network*. *C_network* is used to carry coherence messages (multicast or broadcast messages such as barrier messages or cache coherence messages (mainly due to false sharing)) between the SMs (L1-to-L1), whereas the *NC_network* is used to carry the non-coherence messages (L1-to-L2 and L2-to-L1).

C_network: Insight:SWMR says that the coherence messages are broadcast/multicast messages and hence SWMR links are the best suited for such messages. Consequently, we have used an SWMR based crossbar to connect the *SM_stations* with each other. This network contains 8 SWMR links with each link composed of 1 waveguide, carrying 64 wavelengths. All the coherence traffic is routed through this network.

NC_network: For the *NC_network*, we propose to use MWSR optical links. The reason is the requirement of high bandwidth for carrying the non-coherence traffic. These optical links pass through all the optical stations and thus provide an all-to-all communication framework. Each link in the *NC_network* is assumed to be carrying 128 wavelengths: it is 2 waveguides wide. Moreover,

[3] *LLC* → Last level cache

we consider a double data rate scheme in which optical stations are allowed to transmit signals at both edges of the clock. Thus, a single cache line (64 bytes) can be sent in 2 clock cycles. With a 5GHz clock, each wavelength is capable of providing 10Gb/s bandwidth, yielding 3.2TB/s bandwidth for the *NC_network*.

5.3.2 Arbitration. In the *NC_network*, multiple stations are allowed to write on a single link. Thus, to provide mutual exclusion, we require arbitration at the sender side. We propose to use the distributed token-based arbitration scheme [59]. In this scheme, we assume a dedicated waveguide going through all the optical stations, called arbitration waveguide (or token waveguide). This waveguide runs parallel to the data waveguides of *NC_network* in serpentine layout. We propose to inject x (equal to number of data links in the *NC_network*) optical tokens into the token waveguide. Each token in a token waveguide is a single optical pulse at a different wavelength. Token i represents the status of optical link i. The availability of a token (wavelength) in the token waveguide guarantees the availability of that particular optical link. For clarity, we call these tokens as *data_tokens*.

Each optical station is equipped with an array of modulators and detectors attached to the token waveguide (equal to number of tokens), and thus it is capable of diverting any *data_token* from the token waveguide (diverting that wavelength). Whenever a station wants to send data to station m, it first turns on its detector for the *data_token m*. The presence of *data_token m* conveys the information that no other station is using that specific data link. The station grabs this token and then sends the message through the respective optical link. When a station finishes sending its data, it reinjects that specific *data_token* back into the token waveguide (stops diverting the signal with the corresponding wavelength). This arbitration scheme is fair and allocates the tokens in a round-robin fashion. Moreover, if a station wants to send data to multiple receivers, it has the liberty to grab as many *data_token*s as possible. Allowing a station to use multiple tokens in the context of a GPU as our novel contribution. Figure 9 summarizes the arbitration scheme for data links of *NC_network*.

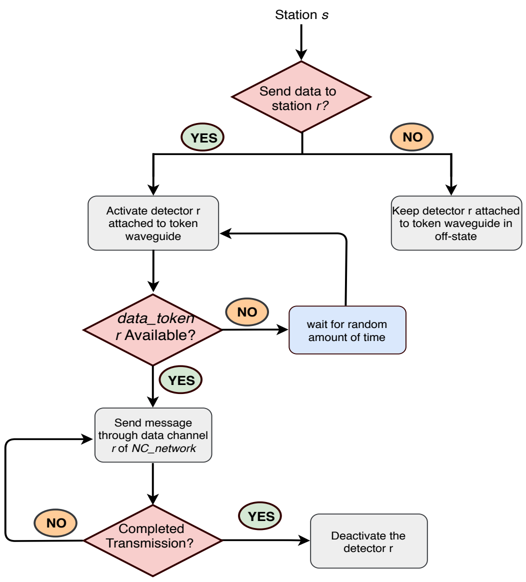
Station s
Send data to station r?
YES
NO
Activate detector r attached to token waveguide
Keep detector r attached to token waveguide in off-state
data_token r Available?
NO
wait for random amount of time
YES
Send message through data channel r of NC_network
NO
Completed Transmission?
YES
Deactivate the detector r

Fig. 9. Flow of operations for accessing data links of NC_network

5.3.3 Power Delivery Network. The off-chip laser source generates monochromatic optical power (at 1550 nm). To decrease insertion loss we use special tapered waveguides [27] to couple light into the chip. Inside the chip, the incoming power is divided into 64 equally spaced wavelengths (between 1450nm to 1650nm) at each station in order to enable DWDM based communication using a comb splitter (see Section 3.1). The power is routed to all the optical stations inside the chip using serpentine-shaped power waveguides running parallel to the data waveguides of the *NC_network* as shown in Figure 7.

In our design we propose to use 16 power waveguides. Additionally, we allow the optical stations to share the available optical power. Thus, the optical station has full freedom to divert power from any power waveguide contingent on the fact that it is not being used. Note that since multiple stations might express interest for the same power waveguide arbitration is necessary; we propose to use the same token based mechanism. We inject additional *power_tokens* (one token per waveguide) in the token waveguide. Note that this is in addition to the tokens inserted into the *NC_network* to keep track of the data waveguides. Using separate power and data tokens is a novel contribution, and given the bandwidth requirements of GPUs it is necessary.

5.3.4 Power Sharing and Data transmission. We need to deliver power via power waveguides to three kinds of waveguides: regular data waveguides, the token waveguide (for arbitration tokens), and the prediction waveguide (will be discussed in Section 6). The role of the prediction waveguide is to collate network activity predictions across all the stations.

Inside the chip we divide the optical power into $(n + 2)$ units. n units are sent through n power waveguides (for data transmission), 1 unit goes to the token waveguide, and 1 unit goes to the prediction waveguide. This 1 unit of power is transmitted on the prediction waveguide after splitting it into 16 equispaced wavelengths using a comb splitter.

The power diverted to the token waveguide is split into 32 equally spaced wavelengths. We can thus send 32 tokens (1 per wavelength). We send 16 data tokens and 16 power tokens. Figure 10 summarizes this discussion and shows the infrastructure for distributing power across the waveguides.

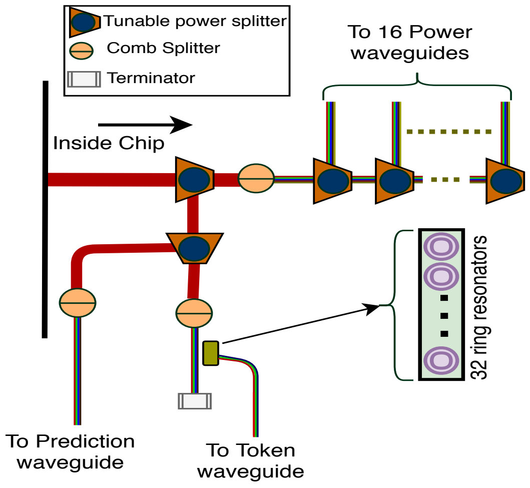

Fig. 10. Power distribution sub-network

To send data, an optical station has to grab at least one *power_token* from the token waveguide. After getting *power_token i*, the station is allowed to divert power from the i^{th} power waveguide.

If it is a coherence message then the optical station needs to use the *C_network*, else it needs to use the *NC_network*. Subsequently, the optical station needs to obtain a *data_token* for the data waveguide corresponding to the receiver, and then send the data. If any of the tokens (power or data) is not available, then the station waits tries again by following an exponential back-off scheme. Figure 11 shows the complete flow of operations. Note that a station is allowed to grab multiple power and data tokens such that it can send multiple messages in parallel.

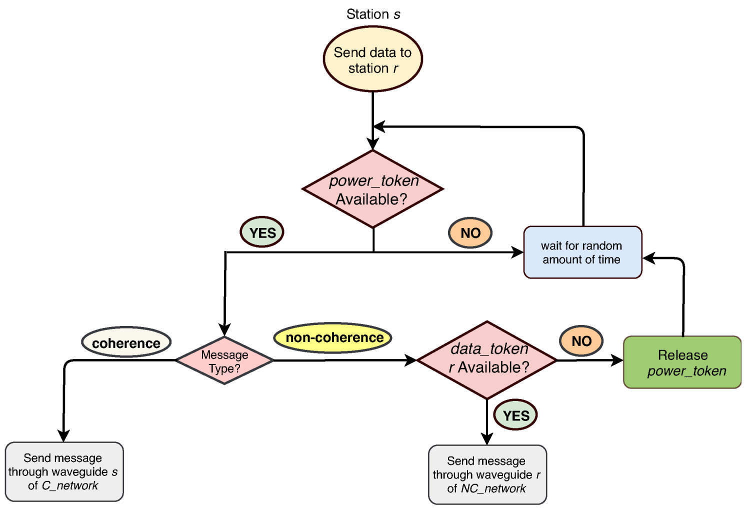
Station *s*
Send data to station *r*
power_token Available?
YES
NO
wait for random amount of time
coherence
Message Type?
non-coherence
data_token r Available?
NO
Release *power_token*
YES
Send message through waveguide *s* of *C_network*
Send message through waveguide *r* of *NC_network*

Fig. 11. Flow of operations

5.4 Optically Connected Memory (OCM)

As is the standard practice, we propose to use optical links to connect on-chip memory controllers with the off-chip DRAM memory modules. Each memory controller is connected to its off-chip memory module by a high-speed optical link that is composed of 4 waveguides (2 in each direction). Each waveguide carries 64 wavelengths using WDM. We assume to send data on both edges of the clock. Considering 5Gb/s wavelength signalling, each waveguide yields 80GBps bandwidth and one such off-chip interconnect link (4 waveguides) is capable of providing 320GBps bandwidth.

6 LASER POWER MANAGEMENT: *PS_GPUOPT*.

6.1 Overview

The standard approach used to decrease the laser power consumption in PNoCs is: ❶ divide the total execution time into fixed-size durations called *epochs*, ❷ predict the laser power requirement of the next epoch (prediction phase), and then ❸ modulate the off-chip laser (reconfiguration phase). Our solution, *PS_GPUOPT*, works on similar lines. However, it is a bespoke scheme for GPUs.

An entire *epoch* is divided into three separate phases: transmission phase, prediction phase, and reconfiguration phase. In the transmission phase, the optical stations are allowed to send messages to each other by diverting available power from the power waveguides. In the prediction phase, the optical stations send their statistics to the *laser controller* through a prediction waveguide. The *laser controller* collates the statistics from all the optical stations and then predicts the amount of traffic required in the next epoch. In the reconfiguration phase, the laser controller dynamically tunes the off-chip laser array. It also reconfigures the split-ratio of on-chip tunable optical power splitters.

6.2 Prediction Mechanism

As described in Section 5.2 there are two types of optical stations: *SM_stations* and *LLC_stations.* We propose to use separate prediction mechanisms for *SM_stations* and *LLC_stations.* The main reason stems from **Insight:Flexible** and **Insight:Rigid**. We choose to use a more restricted predictor (called *Restr_Pred*) for *SM_stations* and a flexible predictor (called *Flex_Pred*) for *LLC_stations.*

6.2.1 Restr_Pred. Every *SM_station* uses the *Restr_Pred* module to predict the laser power requirement in the next epoch. *Restr_Pred* uses a multivariable function (Ψ) to make its prediction. This function takes three inputs : messages received in the current epoch ($\mathcal{M}_{\mathcal{R}}$), messages sent in the current epoch ($\mathcal{M}_{\mathcal{S}}$), and the waiting time of a station ($\mathcal{W}$), and produces a 1-bit output to be sent to a *Laser Controller (L_Cntrlr)* at the end of every epoch. The output indicates if power is required or not in the next epoch. Ψ is defined as follows:

$$\Psi(\mathcal{M}_{\mathcal{R}},\mathcal{M}_{\mathcal{S}},\mathcal{W}) = \begin{cases} 1 & \underbrace{(\mathcal{M}_{\mathcal{R}} \geq R_T \wedge \mathcal{M}_{\mathcal{S}} \leq \alpha * R_T)}_{F_1} \vee \underbrace{(\mathcal{W} \geq W_T)}_{F_2} \\ & \vee \underbrace{(\mathcal{M}_{\mathcal{R}} \leq R_T \wedge \mathcal{M}_{\mathcal{S}} < \alpha * M_R)}_{F_3} \\ 0 & otherwise \end{cases} \tag{1}$$

In this function, R_T, and W_T are threshold values, α and β are constants (all empirically determined). For our prediction scheme, we incorporate three hardware counters [14] ($\mathcal{M}_{\mathcal{R}}$, $\mathcal{M}_{\mathcal{S}}$, and $\mathcal{W}$) inside each optical station to collect these metrics. Along with good empirical results, the intuitions for this formula are as follows.

F_1 means that if a station has received a lot of messages, and sent a few, it is most likely to transmit in the next epoch because it needs to send responses. F_2 suggests that if a lot of messages have been waiting, then they will be sent in the next epoch, and finally F_3 means that if a station has been very quiescent in the current epoch it has a higher likelihood of being active in the next epoch. Note that these are GPU specific observations and do not hold for other platforms.

6.2.2 Flex_Pred. Akin to *SM_stations LLC_stations* also send a 1-bit prediction to the laser controller based on the following rules.

$$\zeta(\mathcal{M}_{\mathcal{R}},\mathcal{P}_{\mathcal{E}},\mathcal{W}) = \begin{cases} 1 & \underbrace{(\mathcal{M}_{\mathcal{R}} \geq \alpha * R_T)}_{G_1} \vee \underbrace{(\mathcal{W} \geq \alpha * W_T)}_{G_2} \\ & \vee \underbrace{(\mathcal{P}_{\mathcal{E}} \geq \beta * \mathcal{M}_{\mathcal{R}})}_{G_3} \\ 0 & otherwise \end{cases} \tag{2}$$

Here, $\mathcal{P}_{\mathcal{E}}$ is the number of pending events at the *LLC_station* in the current epoch. The intuition for G_1 is similar to F_1 (see Equation 1), and that for G_2 is similar to that of F_2. G_3 considers the ratio $\mathcal{P}_{\mathcal{E}}/\mathcal{M}_{\mathcal{R}}$ and predicts a '1' if it is more than a threshold, β. This means that there are more events at the station other than the expected number of responses to reads, and these *events* need to be sent in the next epoch.

6.2.3 Laser Controller (L_Cntrlr). At the end of every epoch, *SM_stations* and *LLC_stations* send their 1-bit recommendations to the *L_Cntrlr* through a separate waveguide called the *prediction waveguide.* This process is initiated 8 cycles before the end of the epoch.

In the reconfiguration phase, the first job of *L_Cntrlr* is to collect 16 1-bit predictions sent by the 16 optical stations. The *L_Cntrlr* adds the eight 1-bit predictions received from the 8 *SM_stations* to calculate the amount of power required by the *SM_stations* in the next epoch. Let this sum be equal to v; if the sum was 8, we set v to 7 such that it remains within 3 bits. We subsequently use the eight 1-bit predictions received from the 8 *LLC_stations* to index a 256-entry table called the Power Table (PT). Each entry of the PT stores the predicted number of power units that will be used by the *LLC_stations* in the next epoch. Let the value read from the table be w (3 bits). We shall explain the reason for different prediction schemes for SMs and LLC banks after explaining how the PT table is updated.

We update the PT at the end of every epoch. Let *PT_index_prev* denote the 8-bit number created using the eight 1-bit predictions received by the *L_Cntrlr* in the previous epoch. Let P be the optical power used in the current epoch and S be the optical power units required in the next epoch (sum of 1-bit predictions), then the *PT* table is updated as follows:

$$PT[PT\_index\_prev] = \begin{cases} P+1 & S \geq P \\ P & P/2 \leq S < P \\ P-1 & default \end{cases} \tag{3}$$

For the SMs ($P = S$), which means that the system reacts immediately and modulates the power requirements. L1 miss rates are more frequent, and thus faster reactivity is beneficial. Additionally, if we underestimate the power, the additional latency will not affect us much (see Section 4.3.3). However, in the case of LLC banks, we make the system far less reactive particularly for reducing the number of tokens. We always prefer to have the same number of tokens (or more) even if the expected number reduces by up to 50% because we wish to prioritize this traffic from the point of view of bandwidth and latency.

Reconfiguration: Finally, the *L_Cntrlr* adds the two 3-bit numbers (v and w) to determine the number of lasers to be turned on in the off-chip laser array. This 4-bit sum is sent to the off-chip laser array for turning on the required number of lasers in the next epoch (see Section 3.1). We assume that the off-chip laser is connected using a fast optical link and it is located at a distance of $2cm$ from the chip. Thus, it takes less than a cycle to reach the off-chip light source.

In addition to tuning the off-chip laser source the on-chip optical power splitters also need to be tuned. Each splitter needs to be tuned to a new split ratio. We use 16 cascaded tunable optical power splitters. Thus, depending upon the amount of predicted power, the splitters are tuned accordingly. In our configuration, the off-chip laser provides 16 different power values, and for each value, the splitter is to be tuned to a different split ratio (6-bit). For efficient tuning we maintain a 16-entry tuning table (TT) at the *L_Cntrlr*. The TT is indexed by the 4-bit sum calculated by the *L_Cntrlr*. Each entry stores the 16 split ratios for the 16 splitters in the network. We require $(16 \times 16 \times 6)/8 = 192$ Bytes to store this table.

Table 1 summarizes the entire scheme with the number of cycles required for different operations in the prediction and reconfiguration phases. In an entire 1000-cycle epoch less than 10 cycles are required for the prediction and reconfiguration phases, and out of these the network is made to halt for less than 4 cycles. Thus, the overhead is less than 0.4% for a 1000 cycle epoch.

7 EXPERIMENTAL METHODOLOGY

Simulated System : Table 2 lists the architectural parameters of the GPU used in our design. The system has 64 SMs where each SM has 32 SPs and 8 SFUs. Additionally, each SM includes an instruction cache, private L1 cache, constant cache, and shared memory. Moreover, the GPU has a shared L2 cache divided into 8 banks, with each bank containing a separate memory controller.

Operation	Cycles
A. Prediction Phase (End of the epoch)	
Calculate the Ψ and ζ functions.	1
Send recommendations (1-bit) to the *L_Cntrlr**.	1
L_Cntrlr collates the recommendations.	1
L_Cntrlr calculates the tuning power.	2
Access the tuning table and calculate the split ratios of all the splitters.	1
B. Reconfiguration Phase (Start of the epoch)	
Reconfigure the off-chip laser and on-chip splitters (parallel activity).	1
Tune the laser array, and optical splitters (parallel activity)*	2
Total Cycles	9
* Network inactive	

Table 1. Algorithm for laser power scaling

Table 3 lists the optical parameters of different optical components used in our design. These are standard parameters that have been taken from published prototypes, and they are also mutually compatible.

Parameter	Value	Parameter	Value
GPU configuration			
Technology	16nm FinFET	Die Size	$600mm^2$
Clock Frequency	1400MHz	# SMs	64
Per SM configuration			
# SPs	32	# LSUs	16
# SFUs	8	# Registers	32678 (32 bits)
L1 Cache	32 KB 4-way	I Cache	8KB 2-way
Constant Cache	16KB 2-way		
LLC configuration			
# L2 Banks	8	Size Per Bank	512KB
Associativity	8	Line Size	128 Byte
Global Memory			
# Memory Controllers	8	Interface	HBM2 DRAM
Optical NoC			
Station Queue size		16	
Electrical NoC			
Topology	Mesh	Routing Algo.	X-Y
Link Traversal	1 cycle	Router Delay	2 cycles
Flit-size	256-bit	Link data rate	10 Gb/s

Table 2. Architectural parameters

Parameter	Value	Parameter	Value
Component configuration			
Wavelength	1550nm	Waveguide Width	$0.5\mu m$
Waveguide Thickness	$0.2\mu m$	Photodetector power	$36\mu W$
Propagation speed in fibres	5ps/mm	Propagation speed in waveguides	11ps/mm
Optical Loss			
Coupling Loss	50%	Waveguide Loss	-0.274dB/cm
Bending Loss	$-0.005dB/90^\circ$	Splitter Loss	-0.36dB
Photodetector Loss	-0.1dB	Crossing Loss	-0.05dB
OFF-state ring loss	-0.005dB	ON-state ring loss	-0.6dB

Table 3. Optical parameters [5–7, 50, 60]

Evaluation Methodology : To evaluate our design, we use a cycle-accurate GPU simulator *GPUTejas* (validated with native hardware [38]). It includes configurable electrical NoC models. We extended the simulator and included models for photonic buses and then implemented our final optical topology.

The applications evaluated in this work were taken from the popular GPU benchmark suite Rodinia [12]. The selected subset of applications represent diverse workloads featuring both memory-intensive and compute-intensive benchmarks. We were not able to simulate a few benchmarks because they did not compile with CUDA 6.0.

The simulator uses the Orion 2 [29] and McPAT [33] tool to compute the power consumed by an electrical NoC. For an optical NoC, the power consumed is calculated analytically based on the standard model proposed by Joshi et al. [28]. In addition, we use Cacti 6.5 [57] to calculate the power and area associated with the additional memory blocks used in our design.

Baseline electrical and optical topology :

Table 2 lists the parameters considered for the electrical NoC, *EMESH*. For a baseline optical topology, we choose the state-of-the-art scheme proposed by Ziabari et al. [65]. We call this scheme *Prior_Opt*. It uses SWMR and MWSR optical networks for L2-to-L1 and L1-to-L2 communication respectively. It does not use any laser modulation technique.

8 EXPERIMENTAL RESULTS

8.1 Electrical Vs Optical

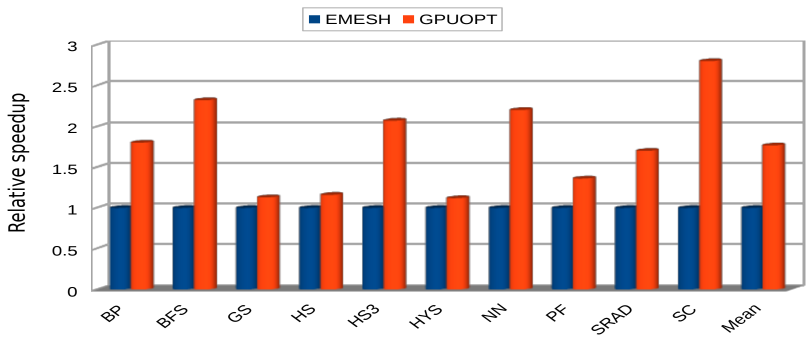

Fig. 12. Electrical vs optical networks

In Figure 12, we compare the relative performance of *EMESH* with *GPUOPT*. We observe from the plot that *GPUOPT* on an average performs 1.8X better than the *EMESH*. In the case of memory-intensive benchmarks such as *BFS*, *HS3*, *NN*, and *SC*, *GPUOPT* performs 2.4X better than the electrical counterpart. The reason stems from the high bandwidth and low latency characteristics of the PNoC. In the case of benchmarks with very little NoC activity such as *GS*, *HS*, and *HYS*, the ENoC performs at par with the PNoC.

Additionally, we have also compared the relative performance of our optical scheme with the adaptive routing [42], bufferless routing [43], and priority-aware [39] electrical NoCs. The results are given in Figure 13. It is clear from the graph that due to the high bandwidth provided by *GPUOPT* (due to optical links), it resulted in a 1.95X, 2.05X, and 1.8X improvement in performance as compared to the bufferless, adaptive, and priority-aware NoC schemes respectively.

Summary: We can conclude that the PNoC performs much better than the ENoC in the case of benchmarks with sufficient NoC activity.

8.2 Analysis

Table 4 analyzes the effect of the laser modulation technique (*PS_GPUOPT*) on the overall performance and power consumption of the system. We observe that by incorporating the laser modulation scheme in *GPUOPT*, there is a 49% increase in the average wait time (sometimes power

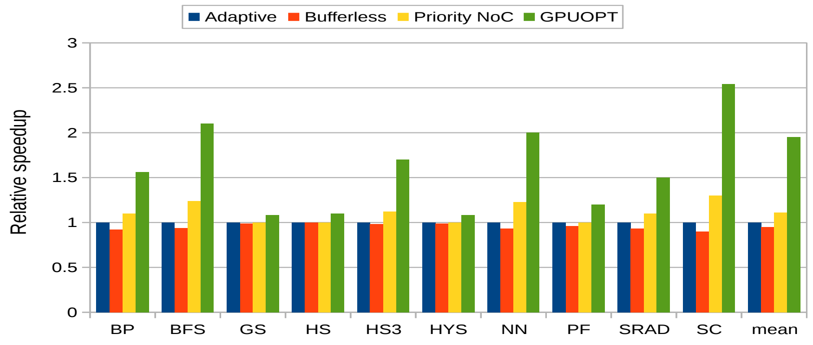

Fig. 13. Relative performance comparison across different electrical NoCs

Benchmarks	Average wait time per request (cycles)			Average # of tokens per epoch	
		$\alpha = 0.5$ $\beta = 0.25$	$\alpha = 1.0$ $\beta = 0.5$	$\alpha = 0.5$ $\beta = 0.25$	$\alpha = 1.0$ $\beta = 0.5$
	A	B	B	B	B
BP	12.20	16.30	16.92	6.20	5.83
BFS	18.15	23.27	23.81	8.13	7.39
GS	2.20	3.31	4.37	1.83	1.64
HS	1.60	2.79	3.61	1.42	1.22
HS3	13.30	17.10	18.33	8.41	7.20
HYS	1.18	1.86	2.47	1.17	1.07
NN	13.40	18.52	20.15	8.27	7.31
PF	2.52	4.31	5.72	2.47	1.87
SRAD	6.47	9.20	10.83	6.74	5.17
SC	19.20	26.71	28.06	7.38	6.68
Mean	**9.02**	**12.34**	**13.43**	**5.20**	**4.54**

Table 4. Analysis of the wait time and the number of tokens (A = *GPUOPT*, B = *PS_GPUOPT*)

is not available). However, it also significantly reduces the number of tokens used per epoch, effectively reducing the power usage.

Subsequently, we simulated the system by varying the constants, threshold values, and epoch size in our scheme and chose the optimal values in the final evaluation results. In Table 4, we compared the effect of the constants, α and β (closest competitors) on the average wait time and the number of tokens sent into the system. We observe that the configuration with $\alpha = 0.5$, and $\beta = 0.25$ is the most optimal configuration. Also, our simulation results depict that the 1000 cycle epoch, $R_T = 128$, and $W_T = 1000$ is the best configuration in terms of system performance and power consumption. Thus, we have assumed the same values in our final evaluation.

8.3 Comparison with Prior Work on PNoCs

8.3.1 Performance Comparison. Figure 14 compares the performance of the three different optical NoC configurations for 10 workloads from the Rodinia benchmark suite. Form the plot, we conclude that *GPUOPT* is the best configuration performing 1.2X better than the *Prior_Opt*. This is because *GPUOPT* allows all the optical stations to share the available optical power. Besides, separating the coherence and non-coherence messages further adds to the performance of the overall system. Moreover, the reduction in the overall performance of the system for the *PS_GPUOPT* configuration is because of the laser modulation scheme. In *GPUOPT* the optical power is available all the time since it does not use any laser modulation technique and hence performs better than *PS_GPUOPT*.

Additionally, we find that in the case of benchmarks with a higher number of memory transactions (L2 and main memory), *GPUOPT* shows a greater improvement in performance. However, for compute-intensive benchmarks, *GPUOPT* performs roughly the same as *Prior_Opt*. Overall,

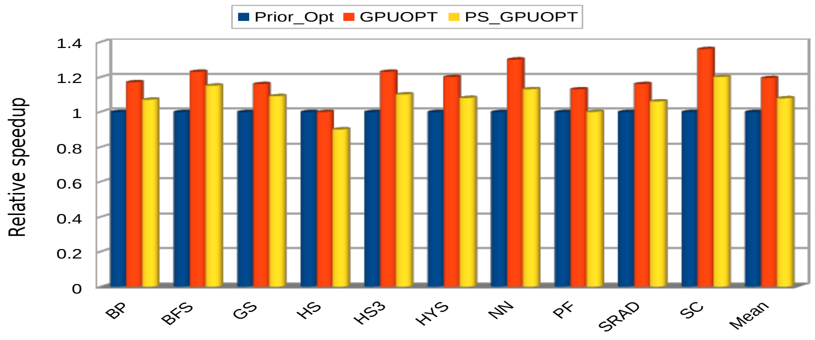

Fig. 14. Performance comparison

GPUOPT is the best configuration in terms of performance performing 17% and 11% better than *Prior_Opt* and *PS_GPUOPT* respectively.

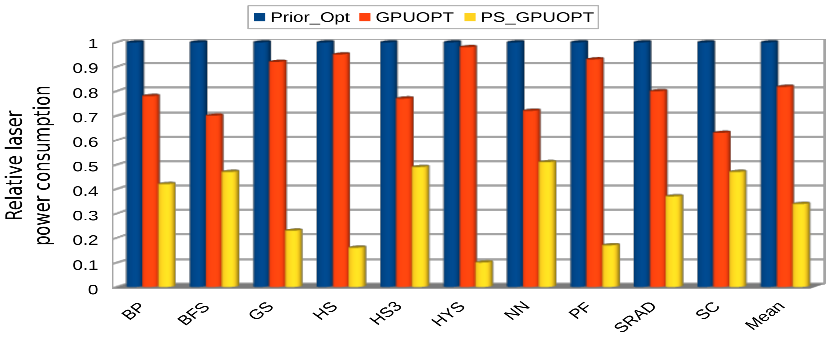

Fig. 15. Relative laser power consumption

8.3.2 Laser Power Consumption: To show how our laser modulation scheme affects the overall laser power consumption, we used the analytical model proposed by Joshi et al. [28] to calculate the total laser power consumption. The various optical losses that occur during the message transmission are modeled according to the values given in Table 3. Based on these values, we layout our waveguides and found out the amount of power required per wavelength in order to activate the farthest detector.

Figure 15 compares the laser power consumption across different configurations. The plot shows that *Prior_Opt* consumes the maximum amount of power, followed by *GPUOPT*. *PS_GPUOPT* consumes the least power as it modulates the off-chip laser based on our proposed prediction scheme. The maximum improvement is shown in the case of workloads such as *HS*, *HYS*, and *PF*. It is because in these programs the entire application fits inside the L1 cache and hence there is very low NoC activity. Because of lower NoC activity, our prediction scheme keeps most of the lasers in the off-chip laser array in switched off. This decreases the laser power consumption significantly.

To summarize, incorporating our prediction scheme in the proposed PNoC decreases the laser power consumption by nearly 59%. As compared to *Prior_Opt*, *PS_GPUOPT* results in a 67% reduction in laser power consumption. The lower laser power consumption in *GPUOPT* as compared to *Prior_Opt* is attributed to its ability to allow on-chip optical stations to share the available optical power.

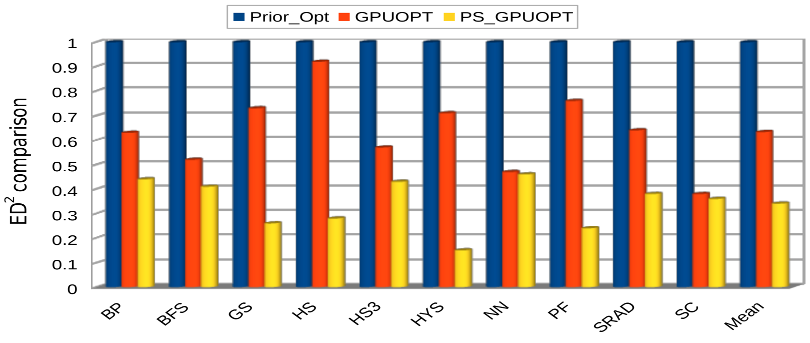

Fig. 16. Energy-Delay-Square product comparison

8.3.3 ED^2 Comparison: In Figure 16 we compared the energy-delay-square (ED^2) product of the three optical configurations. Here, *energy* is the overall energy of the system and *delay* is the total execution time. The ED^2 product is the standard metric used to compare systems that have different delays and power consumption values; the lower the ED^2, the more power-efficient is the system. We again observe that *PS_GPUOPT* is the best configuration. As compared to *Prior_Opt* and *GPUOPT*, it results in a 65% and 29% reduction in ED^2 respectively. The lower ED^2 in the case of *GPUOPT* as compared to *Prior_Opt* is attributed to its higher performance, whereas in the case of *PS_GPUOPT* the decrease is due to a greater reduction in the laser power consumption.

8.4 Analysis of the Prediction Scheme

In any network where the optical stations are allowed to share the optical power, it is not possible to directly determine the prediction accuracy. However, in such scenarios, we indirectly try to ascertain the prediction accuracy. In our scheme, we attempt to indirectly determine the false positives and false negatives of our prediction scheme. Having a higher number of false negatives will decrease the performance of the system, whereas a higher number of false positives shall increase the laser power consumption.

From Figure 14 we observe that by using our prediction scheme, it results in only a 11% reduction in performance as compared to a scheme that does not use any laser modulation technique (*GPUOPT*). Thus, it implies that the effect of false negatives is modest. Second, Figure 15 shows that our prediction scheme results in a 59% reduction in the laser power consumption indicating that our prediction scheme is associated with very few false positives. Based on such indirectly observed metrics, we may conclude that our predictor has an acceptable accuracy.

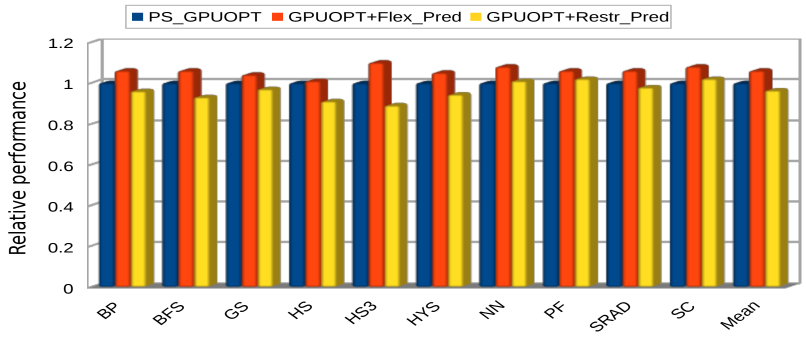

Fig. 17. Relative speedup

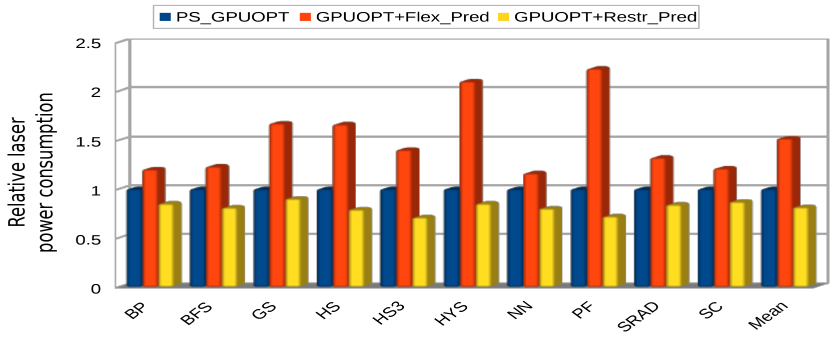

Fig. 18. Relative laser power consumption

To show the effect of using different predictors for *SM_stations* and *LLC_stations*, we derived two new configurations from *GPUOPT*. In one configuration we used only *Flex_Pred* in *GPUOPT*, and in the other configuration we used only *Restr_Pred*. Both these configurations are compared with *PS_GPUOPT*. The results are shown in Figure 17 and 18. We observe from the plots that by using only the *FLex_Pred* scheme, there is a 6% improvement in performance as compared to *PS_GPUOPT*. However, the same scheme increases the laser power consumption by 34% as compared to *PS_GPUOPT*. The ED^2 is 20.6% more. Likewise, using only *Restr_Pred* results in a 4% decrease in performance and a 19% decrease in laser power consumption as compared to *PS_GPUOPT*. The ED^2 also increases by 18.1%. Thus, we can conclude from these results that by using only one kind of predictor, only one parameter improves at the significant expense of the other. There is a net increase in the ED^2, and thus we need to have both the types of predictors.

9 RELATED WORK

A large amount of work has been done in the field of designing efficient photonic on-chip networks for multicore systems. These topologies include mesh, torus, butterfly, multilayer, and crossbar based topologies. However, the area of NoCs for GPUs (both electrical and optical) is *very sparse*.

9.1 Electrical NoCs for GPUs

Bakhoda et al. [1, 3] has evaluated how the performance of a GPU varies across different micro-architectures and electrical NoC designs for different benchmarks. They also analyzed how router latencies affect the GPU performance. Similarly, the authors of [64] analyzed the performance and power consumption of GPU across different symmetric and assymmetric electrical NoCs.

The authors of [2] leveraged the many-to-few traffic patterns in GPUs and proposed the scheme of alternating the full routers with half routers in congested area in order to reduce the NoC area.

9.2 Optical NoCs for GPUs

Lee et al. [31] provided a thorough survey regarding the effect of different network parameters on the performance of a CPU-GPU system similar to our analyses. However, our analysis focuses more on the relative insensitivity of the latency to performance, in stark comparison to the throughput. We additionally focus on assessing scalability, and classifying the nature of the on-chip traffic on a GPU.

Goswami et al. [24] proposed a 3D optical topology that uses an MWMR based crossbar to connect the shader cores with the memory controllers. On similar lines Ziabari et al. [65] proposed an SWMR and MWSR based hybrid crossbar to connect the shader cores with the last level cache banks. The main difference between our work and these previous proposals is that they have not taken into account the benchmark specific characteristics for designing the topologies resulting in increased contention. Additionally, they have not considered the issue of static power consumption, which is one of the major bottlenecks in the deployment of on-chip photonic networks.

10 CONCLUSION

Modern GPU workloads are very sensitive to the bandwidth of the on-chip network, and relatively insensitive to its latency as long as buffer space is not an issue. They thus stand to significantly gain if we use photonic networks given their high bandwidth, ultra-low latency, and low buffer space requirements. However, in such networks ensuring high performance, and minimizing the laser power loss is a challenging problem.

In this paper we propose a novel photonic network called *GPUOPT* that divides SMs and LLC banks into separate clusters, and has two distinct sub-networks: one for coherence messages, and the other for the rest of the messages. They are architected differently. Furthermore, to decrease the high static power consumption in PNoCs, we propose a GPU specific prediction mechanism for modulating the off-chip light source. By using these techniques, we were able to improve the performance of a 64-SM GPU by 45% as compared to a state-of-the-art electrical network. Moreover, as compared to a state-of-the-art photonic network, our design resulted in a 67% decrease in laser power consumption, thereby reducing ED^2 by 65% for workloads from the *Rodinia* benchmark suite.

REFERENCES

[1] A. Bakhoda, J. Kim, and T. M. Aamodt. 2010. On-chip network design considerations for compute accelerators. In *2010 19th International Conference on Parallel Architectures and Compilation Techniques (PACT)*.

[2] Ali Bakhoda, John Kim, and Tor M. Aamodt. 2010. Throughput-Effective On-Chip Networks for Manycore Accelerators. In *Proceedings of the 2010 43rd Annual IEEE/ACM International Symposium on Microarchitecture (MICRO âĂŹ43)*. IEEE Computer Society, USA. https://doi.org/10.1109/MICRO.2010.50

[3] A. Bakhoda, G. L. Yuan, W. W. L. Fung, H. Wong, and T. M. Aamodt. 2009. Analyzing CUDA workloads using a detailed GPU simulator. In *2009 IEEE International Symposium on Performance Analysis of Systems and Software*. 163–174.

[4] Janibul Bashir, Eldhose Peter, and Smruti R Sarangi. 2019. BigBus: A Scalable Optical Interconnect. *ACM Journal on Emerging Technologies in Computing Systems (JETC)* 15, 1 (2019), 8.

[5] Janibul Bashir, Eldhose Peter, and Smruti R. Sarangi. 2019. A Survey of On-Chip Optical Interconnects. *ACM Comput. Surv.* 51, 6 (Jan. 2019), 115:1–115:34.

[6] Janibul Bashir and Smruti R Sarangi. 2017. NUPLet: A Photonics Based Multi-Chip NUCA Architecture. In *2017 IEEE 35th International Conference on Computer Design (ICCD)*. IEEE.

[7] Janibul Bashir and Smruti Ranjan Sarangi. 2019. Predict, Share, and Recycle Your Way to Low-power Nanophotonic Networks. *ACM Journal on Emerging Technologies in Computing Systems (JETC)* 16, 1 (2019), 1–26.

[8] Janibul Bashir, Khushal Sethi, and Smruti R Sarangi. 2019. Power efficient photonic network-on-chip for a scalable GPU. In *Proceedings of the 13th IEEE/ACM International Symposium on Networks-on-Chip*. 1–2.

[9] J-R Burie, G. Beuchet, M. Mimoun, P. Pagnod-Rossiaux, B. Ligat, JC Bertreux, J-M Rousselet, J. Dufour, P. Rougeolle, and F. Laruelle. 2010. Ultra high power, ultra low RIN up to 20 GHz 1.55 μm DFB AlGaInAsP laser for analog applications. In *OPTO*. International Society for Optics and Photonics, 76160Y–76160Y.

[10] Rodolfo E Camacho-Aguilera, Yan Cai, Neil Patel, Jonathan T Bessette, Marco Romagnoli, Lionel C Kimerling, and Jurgen Michel. 2012. An electrically pumped germanium laser. *Optics express* 20, 10 (2012), 11316–11320.

[11] Jaime Cardenas, Carl B. Poitras, Jacob T. Robinson, Kyle Preston, Long Chen, and Michal Lipson. 2009. Low loss etchless silicon photonic waveguides. *Opt. Express* (Mar 2009).

[12] Shuai Che, Michael Boyer, Jiayuan Meng, David Tarjan, Jeremy W Sheaffer, Sang-Ha Lee, and Kevin Skadron. 2009. Rodinia: A benchmark suite for heterogeneous computing. In *Workload Characterization, 2009. IISWC 2009. IEEE International Symposium on*. IEEE, 44–54.

[13] Guoqing Chen, Hui Chen, Mikhail Haurylau, Nicholas A Nelson, David H Albonesi, Philippe M Fauchet, and Eby G Friedman. 2007. Predictions of CMOS compatible on-chip optical interconnect. *Integration, the VLSI journal* 40, 4 (2007), 434–446.

[14] Xuning Chen, Li-Shiuan Peh, Gu-Yeon Wei, Yue-Kai Huang, and Paul Prucnal. 2005. Exploring the design space of power-aware opto-electronic networked systems. In *11th International Symposium on High-Performance Computer*

Architecture. IEEE, 120–131.
[15] John Cheng, Max Grossman, and Ty McKercher. 2014. *Professional Cuda C Programming*. John Wiley & Sons.
[16] William J Dally and Brian Towles. 2001. Route packets, not wires: on-chip inteconnection networks. In *Proceedings of the 38th annual Design Automation Conference*. Acm, 684–689.
[17] Luan H. K. Duong, Mahdi Nikdast, Jiang Xu, Zhehui Wang, Yvain Thonnart, Sébastien Le Beux, Peng Yang, Xiaowen Wu, and Zhifei Wang. 2015. Coherent Crosstalk Noise Analyses in Ring-Based Optical Interconnects. In *Proceedings of the 2015 Design, Automation and Test in Europe Conference and Exhibition*. EDA Consortium, San Jose, CA, USA.
[18] M. FAUGERON, M. Chtioui, A Enard, O. Parillaud, F. Lelarge, M. Achouche, J. Jacquet, A Marceaux, and F. van Dijk. 2013. High Optical Power, High Gain and High Dynamic Range Directly Modulated Optical Link. *Lightwave Technology, Journal of* 31, 8 (April 2013), 1227–1233.
[19] Mickaël Faugeron, Michaël Tran, François Lelarge, Mourad Chtioui, Yannick Robert, Eric Vinet, Alain Enard, Joël Jacquet, and Frederic Van Dijk. 2012. High-Power, Low RIN 1.55-Directly Modulated DFB Lasers for Analog Signal Transmission. *Photonics Technology Letters* 24, 2 (2012), 116–118.
[20] M. Faugeron, M. Tran, O. Parillaud, M. Chtioui, Y. Robert, E. Vinet, A. Enard, J. Jacquet, and F. V. Dijk. 2013. High-power tunable dilute mode DFB laser with low RIN and narrow linewidth. *Photonics Technology Letters, IEEE* 25, 1 (2013), 7–10.
[21] Naznin Fauzia, Louis-Noël Pouchet, and P Sadayappan. 2015. Characterizing and enhancing global memory data coalescing on GPUs. In *Proceedings of the 13th Annual IEEE/ACM International Symposium on Code Generation and Optimization*. IEEE Computer Society, 12–22.
[22] J. Fujikata, K. Nishi, A. Gomyo, J. Ushida, I. Tsutomu, H. Yukawa, D. Okamoto, M. Nakada, T. Shimizu, M. Kinoshita, et al. 2008. LSI on-chip optical interconnection with Si nano-photonics. *IEICE transactions on electronics* 91, 2 (2008), 131–137.
[23] Rajib R Ghosh, Janib Bashir, Smruti R Sarangi, and Anuj Dhawan. 2019. SpliESR: Tunable Power Splitter Based on an Electro-Optic Slotted Ring Resonator. *Optics Communications* (2019).
[24] N. Goswami, Z. Li, R. Shankar, and T. Li. 2014. Exploring Silicon Nanophotonics in Throughput Architecture. *IEEE Design Test* 31, 5 (Oct 2014), 18–27.
[25] Michael K Gowan, Larry L Biro, and Daniel B Jackson. 1998. Power considerations in the design of the Alpha 21264 microprocessor. In *Proceedings of the 35th annual Design Automation Conference*. ACM, 726–731.
[26] H. Gu and J. Xu. 2009. Design of 3D Optical Network on Chip. In *2009 Symposium on Photonics and Optoelectronics*.
[27] M. J. Humphrey. 1994. *Calculation of coupling between tapered fiber modes and whispering-gallery modes of a spherical microlaser*. Ph.D. Dissertation. University of Maryland, College Park, Maryland.
[28] Ajay Joshi, Christopher Batten, Yong-Jin Kwon, Scott Beamer, Imran Shamim, Krste Asanovic, and Vladimir Stojanovic. 2009. Silicon-photonic clos networks for global on-chip communication. In *NoCS*.
[29] Andrew B Kahng, Bin Li, Li-Shiuan Peh, and Kambiz Samadi. 2011. Orion 2.0: A power-area simulator for interconnection networks. IEEE.
[30] Leonid Khriachtchev. 2016. *Silicon Nanophotonics: Basic Principles, Present Status, and Perspectives*. CRC Press.
[31] Jaekyu Lee, Si Li, Hyesoon Kim, and Sudhakar Yalamanchili. 2013. Design Space Exploration of On-chip Ring Interconnection for a CPU-GPU Heterogeneous Architecture. *J. Parallel Distrib. Comput.* 73, 12 (Dec. 2013).
[32] Jacob S Levy, Yoshitomo Okawachi, Michal Lipson, Alexander L Gaeta, and Kasturi Saha. 2011. High-performance silicon-based multiple wavelength source. In *CLEO: Science and Innovations*. OSA, CMAA7.
[33] Sheng Li, Jung Ho Ahn, Richard D Strong, Jay B Brockman, Dean M Tullsen, and Norman P Jouppi. 2009. McPAT: an integrated power, area, and timing modeling framework for multicore and manycore architectures. In *MICRO*.
[34] Erik Lindholm, John Nickolls, Stuart Oberman, and John Montrym. 2008. NVIDIA Tesla: A unified graphics and computing architecture. *IEEE micro* 28, 2 (2008).
[35] John DC Little and Stephen C Graves. 2008. Little's law. In *Building intuition*. Springer, 81–100.
[36] Jifeng Liu, Xiaochen Sun, Rodolfo Camacho-Aguilera, Lionel C Kimerling, and Jurgen Michel. 2010. Ge-on-Si laser operating at room temperature. *Optics letters* 35, 5 (2010), 679–681.
[37] LUXTERA. 2001. LUXTERA: Fibre to the Chip. *http://www.luxtera.com/luxtera/products* (2001).
[38] Geetika Malhotra, Seep Goel, and Smruti R Sarangi. 2014. Gputejas: A parallel simulator for gpu architectures. In *High Performance Computing (HiPC), 2014 21st International Conference on*. IEEE, 1–10.
[39] Sumit K. Mandal, Raid Ayoub, Michael Kishinevsky, and Umit Y. Ogras. 2019. Analytical Performance Models for NoCs with Multiple Priority Traffic Classes. *ACM Trans. Embed. Comput. Syst.* 18, 5s (2019). https://doi.org/10.1145/3358176
[40] Rainer Michalzik. 2012. *VCSELs: fundamentals, technology and applications of vertical-cavity surface-emitting lasers*. Vol. 166. Springer.
[41] Jurgen Michel, Jifeng Liu, and Lionel C Kimerling. 2010. High-performance Ge-on-Si photodetectors. *Nature photonics* 4, 8 (2010), 527.

[42] Ming Li, Qing-An Zeng, and Wen-Ben Jone. 2006. DyXY - a proximity congestion-aware deadlock-free dynamic routing method for network on chip. In *2006 43rd ACM/IEEE Design Automation Conference*.

[43] Thomas Moscibroda and Onur Mutlu. 2009. A Case for Bufferless Routing in On-Chip Networks. *SIGARCH Comput. Archit. News* 37, 3 (June 2009). https://doi.org/10.1145/1555815.1555781

[44] Berkeley News. 2015. *http://news.berkeley.edu/2015/12/23/electronic-photonic-microprocessor-chip/* (2015).

[45] John Nickolls and William J Dally. 2010. The GPU computing era. *IEEE micro* 30, 2 (2010).

[46] NVIDIA. 2016. NVIDIA Tesla P100: The Most Advanced Datacenter Accelerator Ever Built Featuring Pascal GP100, the World's Fastest GPU. *Whitepaper* (2016).

[47] John D Owens, Mike Houston, David Luebke, Simon Green, John E Stone, and James C Phillips. 2008. GPU computing. *Proc. IEEE* 96, 5 (2008), 879–899.

[48] Yan Pan, John Kim, and Gokhan Memik. 2010. Flexishare: Channel sharing for an energy-efficient nanophotonic crossbar. In *HPCA*.

[49] Yan Pan, Prabhat Kumar, John Kim, Gokhan Memik, Yu Zhang, and Alok Choudhary. 2009. Firefly: illuminating future network-on-chip with nanophotonics. In *ACM SIGARCH Computer Architecture News*. ACM.

[50] Eldhose Peter, Arun Thomas, Anuj Dhawan, and Smruti R Sarangi. 2015. ColdBus: A Near-Optimal Power Efficient Optical Bus. In *HiPC*.

[51] K. Preston, N. Sherwood-Droz, J. S. Levy, and M. Lipson. 2011. Performance guidelines for WDM interconnects based on silicon microring resonators. In *CLEO*.

[52] Circuits Multi Projects. 2018. Silicon Photonics. *https://mycmp.fr/datasheet/silicon-photonic-ics-si310-phmp2m* (2018).

[53] Xu Qianfan, Manipatruni Sasikanth, Schmidt Brad, Shakya Jagat, and Lipson Michal. 2007. 12.5 Gbit/s carrier-injection-based silicon micro-ring silicon modulators. *Opt. Express* 15 (Jan 2007).

[54] Carl Ramey. 2011. Tile-gx100 manycore processor: Acceleration interfaces and architecture. In *2011 IEEE Hot Chips 23 Symposium (HCS)*. IEEE, 1–21.

[55] Jason Sanders and Edward Kandrot. 2010. *CUDA by example: an introduction to general-purpose GPU programming*. Addison-Wesley Professional.

[56] Jinuk Luke Shin, Kenway Tam, Dawei Huang, Bruce Petrick, Ha Pham, Changku Hwang, Hongping Li, Alan Smith, Timothy Johnson, Francis Schumacher, et al. 2010. A 40nm 16-core 128-thread CMT SPARC SoC processor. In *2010 IEEE International Solid-State Circuits Conference-(ISSCC)*. IEEE, 98–99.

[57] S Thoziyoor, N Muralimanohar, JH Ahn, and NP Jouppi. 2008. Cacti 5.3. *HP Laboratories, Palo Alto, CA* (2008).

[58] A. W. Topol, D. C. L. Tulipe, L. Shi, D. J. Frank, K. Bernstein, S. E. Steen, A. Kumar, G. U. Singco, A. M. Young, K. W. Guarini, and M. Ieong. 2006. Three-dimensional integrated circuits. *IBM Journal of Research and Development* (July 2006).

[59] Dana Vantrease, Nathan Binkert, Robert Schreiber, and Mikko H Lipasti. 2009. Light speed arbitration and flow control for nanophotonic interconnects. In *Microarchitecture, 2009. MICRO-42*. IEEE.

[60] Dana Vantrease, Robert Schreiber, Matteo Monchiero, Moray McLaren, Norman P. Jouppi, Marco Fiorentino, Al Davis, Nathan Binkert, Raymond G. Beausoleil, and Jung Ho Ahn. 2008. Corona: System Implications of Emerging Nanophotonic Technology. In *ISCA*.

[61] Yiyuan Xie, Mahdi Nikdast, Jiang Xu, Wei Zhang, Qi Li, Xiaowen Wu, Yaoyao Ye, Xuan Wang, and Weichen Liu. 2010. Crosstalk noise and bit error rate analysis for optical network-on-chip. In *Proceedings of the 47th Design Automation Conference*. ACM, 657–660.

[62] Li Zhou and Avinash Karanth Kodi. 2013. Probe: Prediction-based optical bandwidth scaling for energy-efficient nocs. In *NOCS*.

[63] Amir Kavyan Ziabari. 2016. Improving the global memory efficiency in GPU-based systems.

[64] Amir Kavyan Ziabari, José L. Abellán, Yenai Ma, Ajay Joshi, and David Kaeli. 2015. Asymmetric NoC Architectures for GPU Systems. In *Proceedings of the 9th International Symposium on Networks-on-Chip (NOCS '15)*. ACM, New York, NY, USA.

[65] Amir Kavyan Kavyan Ziabari, Jose L Abellán, Rafael Ubal, Chao Chen, Ajay Joshi, and David Kaeli. 2015. Leveraging silicon-photonic noc for designing scalable gpus. In *Proceedings of the 29th ACM on International Conference on Supercomputing*. ACM, 273–282.

[66] D. Zydek, N. Shlayan, E. Regentova, and H. Selvaraj. 2008. Review of Packet Switching Technologies for Future NoC. In *ICSENG*. 306–311.