A Low-Latency Centralized Controller for MZI-based Optical Integrated Networks

Felipe Göhring de Magalhães[†]*, Rubana Priti[‡], Mahdi Nikdast[†‡], Fabiano Hessel*,
Odile Liboiron-Ladouceur[‡] and Gabriela Nicolescu[†]
[†]Ecole Polytechnique de Montreal, Canada - *PPGCC/PUCRS, Porto Alegre, Brazil - [‡]McGill University, Canada
Contact: felipe.gohring-de-magalhaes@polymtl.ca

***Abstract*—The design, implementation, and validation of a low-latency look-up-table-based centralized controller (LUCC) is presented. The LUCC is applied to different Mach-Zehnder Interferometer (MZI)-based photonic interconnects under dynamic workloads. We obtain a response time of only one clock cycle when using the LUCC in which the scalability can be up to 64 I/O ports.**

Keywords—Optical Networks, Network Controller, Mach-Zehnder Interferometer

I. Introduction

Optical Integrated Networks (OINs) are already a reality for long-distance communications [1] and their usage for short-distance communications, such as inter-chip communications, have already been proven to be applicable [2][3]. Recently, previous visionary work presented photonic architectures with low power consumption, low insertion loss (7.9 dB for an 8×8 structure) and a power penalty of less than 1 dB [4]. This work brings forward OINs as attractive candidates for high demanding communicating architectures. However, the performance and efficiency of such architectures are constrained by their controllers. The control part has an impact on the OIN's overall performance and a better solution is yet to be found. Current work demonstrated architecture with either long setup time (e.g., circuit switch), or become too complex challenging practical deployment [5][6]. Thus, while centralized controllers have been demonstrated for Mach-Zehnder Interferometer (MZI)-based networks [7], further improvement in their response time is required, for practical deployment of OINs.

This work presents the design, implementation and validation of a centralized controller for MZI-based integrated optical networks with a focus on a low-latency solution. This controller runs at the maximum clock rate. In other words, all connections are set-up within a clock pulse for minimum delay when the switch configuration dynamically changes. In order to do so, the controller is designed relying on look-up tables within a fast algorithm for conflict resolution. This controller is suitable for optical networks with up to 64 I/O ports. It takes advantage of an optimization algorithm to reduce the size of the look-up tables, and thus reduces the memory utilization. As a result, it can run on platforms with limited resources (e.g., FPGAs).

The remainder of this paper is organized as follows: Section II presents an overview of OINs followed by Section III in which the LUCC algorithm is explained. Section IV shows the results as well as the scalability potential of the approach, and finally Section V concludes the paper.

II. Optical Integrated Networks

Optical Integrated Networks (OINs) are architectures composed of basic optical building block devices, such as MZIs, which can be grouped towards different architectures or topologies, each with its own advantages and drawbacks. This work focuses on three different OIN topologies: Beneš [8], MIMTRIC and SF (Straight-Forward). While Beneš is a well known topology, MIMTRIC and SF are in-house topologies providing a wider path availability. A silicon-on-insulator (SOI) MZI-based 4×4 switch was recently characterized with the on-chip insertion loss of the 2×2 MZI building block shown to be approximately 1.6 dB [4]. Figure 1 presents the basic switches structures as well as all three topologies for an 8×8 configuration, which are used for validation purpose.

In addition to optical switches, an electronic part is generally present on the system being responsible for the control. This controller receives requests from input nodes (microprocessors, DSPs, memories) and solves conflicts in order to guarantee messages delivery. Also, the controller is responsible for setting up the optical paths for the cases where switch configuration is needed. Control latency is key to ensure the benefits of optics in therm of high throughput so its design must focus on reducing the impact of its execution on time.

III. Proposed Control Unit

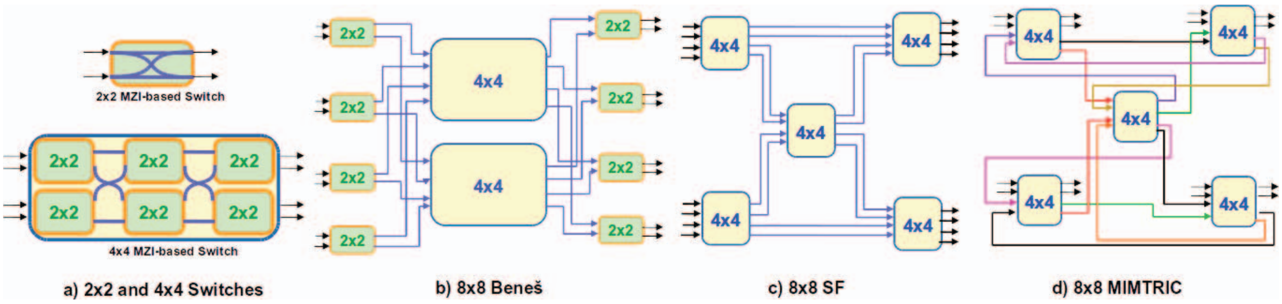
In order to control the system, we opted for a centralized Control Unit decreasing the overall design complexity by providing a systems global view. As a result, LUCC is designed following three steps:

1) In the first step, we reduce the control overhead using an established reduction method for the algorithm [9].

2) In order to minimize the impact on the network efficiency, in the second step, we describe the logic to compute all input requests as fast as possible. To achieve the minimum overhead (one clock cycle to process all inputs), the controller is based on the iSLIP algorithm [10]. Since the iSLIP algorithm requires three cycles for computation, we define an accelerated version of the algorithm, where all requests are treated in parallel, such that the computation is performed within one clock cycle.

3) Finally, in the last step, the message paths for the controller are set. As more than one path can be chosen for every input, several computation possibilities for the controller can exist. To solve this problem, a Shortest Path First (SPF) algorithm is used, and we select the Dijkstra algorithm [11].

978-1-4799-8821-1/15/$31.00 ©2015 IEEE



a) 2x2 and 4x4 Switches b) 8x8 Beneš c) 8x8 SF d) 8x8 MIMTRIC

Fig. 1. (a) MZI-based Switches on 2×2 and 4×4 configurations (b) 8×8 Beneš Architecture (c) 8×8 SF Architecture (d) 8x8 MIMTRIC Architecture.

As it would be complex to calculate a new path for every new request, a pre-calculated table is produced and stored as a look-up table. This can be done by implementing the SPF algorithm and running it offline.

For the SPF algorithm, the network has to be first represented as a series of graphs. Figure 2 presents the graph overview of the 2×2 and 4×4 MZI-based switches, where I/Os are represented as graphs' nodes and connections are represented as edges. Each orange box shows a 2×2 switch and each internal node represents one I/O point. In the purple box, six 2×2 switches are connected to form a 4×4 switch.

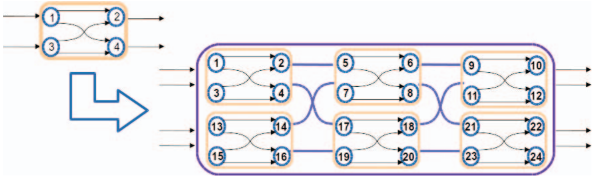


Fig. 2. Connections Graph for 2×2 and 4×4 switches.

Each processed path generates one entry in the table which stores the network configurations. The path allocation tables are then generated for all the three topologies and they can be accessed at run-time. These tables give the computed path to attribute by the controller for each transmission request.

In order for the LUCC to solve conflicts and determine the best path available for each input request, the following steps are taken: **i)** all requested destinations are checked for possible conflicts; **ii)** the target node availability is then ensured as well as the connection status for the request port, so no open connection is left unattended. If the conditions are satisfied, the connection is granted; **iii)** if a conflict is found between transmission requests, a Round-Robin algorithm is used to determine which port will gain access first; **iv)** the path is set once the look-up table has been accessed, and; **v)** the end of communication is verified and an end of transmission acknowledgement signal is set to be true (boolean 1) and then false (boolean 0).

In addition to the LUCC algorithm, another important aspect of the low-latency controller is the look-up table (LUT) generation and optimization. For a small number of I/O ports, the LUT size is negligible, but as the I/O number rises the table gets bigger, to a point where its usage is not practical. The LUT generation has the following steps: **i)** all possible input combinations are found; **ii)** the Dijkstra algorithm is used to compute the best (shortest) path for all the inputs; **iii)** the obtained paths are analyzed and all duplicated situations[1] are excluded, and; **iv)** if a threshold size[2] is verified then the table is reorganized and so are the inputs and outputs. Otherwise, further improvements are applied to the table at the cost of additional logic.

IV. Low-latency centralized controller

A. Validation Platform

The proposed controller was synthesized for an Altera FPGA to ensure its feasibility and it was also simulated in order to guarantee its functionality. All tests were executed over the 4×4 switch architecture as well as on all the three 8×8 topologies. Figure 3 shows one example of the validation platform, where it is possible to see the presence of I/O nodes interacting with LUCC node and with the optical path.

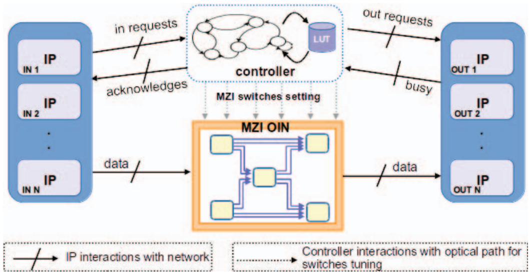


Fig. 3. An example of the validation platform.

B. Results

In order to validate the LUCC, different traffic patterns, such as complement and all-to-all, were applied for each topology to enable different request conditions analysis. Results show LUCC's fast response time in which the delay time is given by the frequency execution of the platform, as the control unit is able to solve requests in one clock cycle. Figure 4 illustrates the one cycle response time of LUCC for any inputs by presenting two scenarios in which it runs at 500 MHz (period of 2 ns).

In the first scenario, marked with yellow boxes, four simultaneous input requests are generated (targeting (**target** signal) outputs 3, 2, 1 and 0) and their access is granted after one clock cycle (on the image, **ack** signals are **1**, one clock

[1] for the cases when, even for different inputs, the switches configurations are the same, thus generating the same table values.

[2] In this context, a threshold size might be defined as any limitation, or constraint, on the final table size. For example, memory utilization on FPGAs can be defined as the upper threshold size.

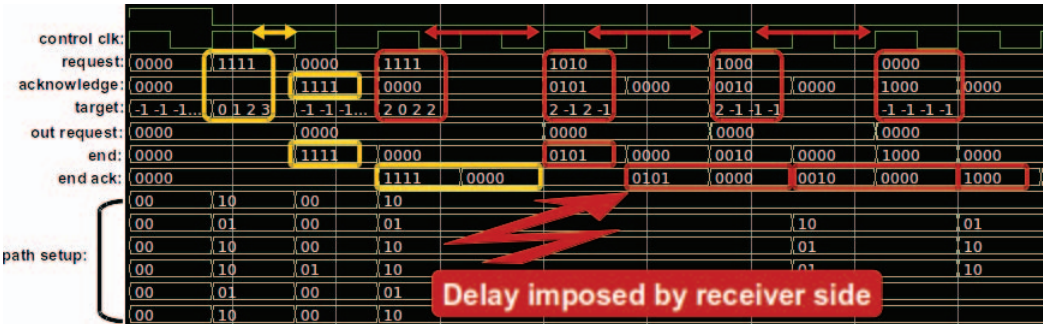


Fig. 4. LUCC timing diagram running at 500 MHz, which shows: input requests (*request*), LUCC grants (*acknowledge*), destination outputs (*target*), LUCC requests for destination outputs (*out request*), end of transmissions (*end*), acknowledges for transmissions ending (*end ack*) and the bits to set-up the MZI switches (*path setup*). On each array of bits, the first bit on the right represents the signal for the I/O zero, the second one the signal for the I/O one, and so on.

cycle after **request** signals are **1**). The yellow arrow shows the period it takes for the LUCC to acknowledge access.

On the second communication scenario, marked with red boxes, four simultaneous input requests are again generated (inputs 0, 1 and 3 are targeting output 2 and input 2 is targeting output 0), but this time with a conflict, as more than one request is targeting output 2. For this case, as previous connections were still open (**end** and **end ack** signals must both be **0** in order to indicate no open connections) it is possible to see the resolution of one conflicted input at a time (**target** -1 means no destination). In this situation, the receiver side takes two cycles to set its connection thus blocking its port. This causes a delay on LUCC granting capabilities, as the destination is blocked until the receiver side compute the end of the transmission. This can be seen on the figure by checking **end** and **end ack** signals.

C. Scalability

Look-up tables are memory hungry and for FPGA-based platforms, this is usually a very important constraint. The size of the table grows exponentially, as Table I presents. In the table, the first column represents the number of I/O ports, the second column holds the number of possible combinations and the third column shows the final number of entries after reduction. For example, two I/O ports have three table entries: 0→1; 1→0, and; 0→1 and 1→0 simultaneously. In this case, no reduction is possible. Still, the table shows values up to ten I/O ports, as a higher number of I/Os generate gigantic quantities (i.e., 16 and 64 I/Os have 2.09×10^{13} and $1{,}26\times10^{56}$ entries, respectively).

TABLE I. LUT GROWING SIZE

I/O	Table Entries	Tables Entries After Reduction (%)
2	3	3 (0%)
3	16	12 (25%)
4	106	61 (45%)
5	778	473 (40%)
6	6598	2875 (56%)
7	63838	29028 (54%)
8	693838	273475 (60%)
9	8361358	3545785 (57%)
10	110557438	45061817 (59%)

Nevertheless, we developed methods based on reduction logic that effectively reduced the table sizes close to 60% by compressing the values. Further reduction is possible by exchanging static paths allocation with wider logic control. Table compression makes it possible to effectively apply this controller in architectures with at most 8 I/O ports. For the cases where a larger number of I/O ports is used, it is possible to replace some of the tables entries with hardware logic, so part of the routes are pre-calculated and stored as look-up tables and the remaining parts are calculated on-the-fly. By replacing static paths with dynamic logic, at the cost of a more expensive hardware implementation, up to 64 I/O ports could be used.

V. CONCLUSION

This work presented the design, implementation and validation of LUCC, a centralized controller for MZI-based Optical Integrated Networks. Results showed a fast response time when employing LUCC in MZI-based optical integrated networks: it takes one clock cycle delay for every request to be computed. Also, the scalability for larger systems was discussed, where the possibilities and limitations of this approach were commented. This type of controller showed to be suitable for small to medium topologies, with up to 64 I/O ports, by the usage of the mentioned methods of offline computing and compression. Furthermore, this approach can be extended to other type of Optical Integrated Networks, using either MZI-based switches or ones exploiting microring Resonators.

REFERENCES

[1] A.F. Benner *et al*, "Exploitation of optical interconnects in future server architectures," *IBM Journal of Research and Development*, July 2005.

[2] A.V. Rylyakov *et al*, "Silicon photonic switches hybrid-integrated with cmos drivers," *IEEE Journal of Solid-State Circuits*, Jan 2012.

[3] B.G. Lee *et al*, "Monolithic silicon integration of scaled photonic switch fabrics, cmos logic, and device driver circuits," *Journal of Lightwave Technology*, Feb 2014.

[4] M.S. Hai *et al*, "MZI-based non-blocking soi switches," in *Asia Communications and Photonics Conference 2014 - paper ATh3A.147.*

[5] A. Biberman *et al*, "Broadband operation of nanophotonic router for silicon photonic networks-on-chip," *IEEE Photonics Technology Letters*, June 2010.

[6] Zheng Li *et al*, "Iris: A hybrid nanophotonic network design for high-performance and low-power on-chip communication," *J. Emerg. Technol. Comput. Syst.*, July 2011.

[7] Fei Lou *et al*, "Towards a centralized controller for silicon photonic MZI-based interconnects," in *Optical Interconnects Conference - paper WD4*, 2015.

[8] V.E. Beneš, "On rearrangeable threestage connecting networks," 1962, pp. 1481–1492, Bell Syst. Tech. J.

[9] S. Le Beux *et al*, "Reduction methods for adapting optical network on chip topologies to 3d architectures," *Microprocess. Microsyst.*, 2013.

[10] Nick McKeown, "The iSLIP scheduling algorithm for input-queued switches," *IEEE/ACM Trans. Netw.*, Apr. 1999.

[11] N. Jasika *et al*, "Dijkstra's shortest path algorithm serial and parallel execution performance analysis," in *Proceedings of the 35th International Convention MIPRO*, May 2012.