Design and Modelling of a Low-Latency Centralized Controller for Optical Integrated Networks

Felipe Göhring de Magalhães, Rubana Priti, Mahdi Nikdast, Fabiano Hessel, Odile Liboiron-Ladouceur, and Gabriela Nicolescu

***Abstract*—Optical integrated network (OIN) is a promising alternative to overcome the restrictions that electrical networks-on-chip (eNoCs) will face in the next generation of multiprocessor integrated systems due to electrical interconnects' physical limitations. OINs present a higher bandwidth and lower power consumption but their full capabilities are curbed by high latency controllers. Control techniques, such as circuit switching, impose a high latency to perform the correct network routing and a better solution must be found in order to fully utilize OINs. In this letter, we have designed and modeled a low-latency centralized controller (LUCC). For validation purposes, we modeled different Mach–Zehnder interferometer (MZI)-based optical interconnects that employ the proposed LUCC. We obtain a response time of only one clock cycle when using the LUCC, even when conflicts are found.**

***Index Terms*—Optical networks, network controller, Mach–Zehnder interferometer, low latency.**

I. Introduction

SINCE the introduction of Multiprocessor Systems-on-Chip (MPSoCs), one of the design's main concerns lies in how the communication between internal components is performed. Systems based on eNoCs tend to provide a good communication performance [1], where the communication management is performed by routers that forward packets inside the network. Besides the gain in the communication capability, eNoCs usually have an improved energy reliability and efficiency and a high re-usability level [2]. However, as the number of possible integrated cores on a single chip continues to increase, metallic interconnects in eNoCs will become a bottleneck due to their high power consumption, limited bandwidth, long latency and poor scalability, leading the International Technology Roadmap for Semiconductors (ITRS) [3] to point out the need for a new technology to overcome such restrictions. In this design context, on-chip optical interconnect is currently considered to be one of the most promising new paradigms.

Optical Integrated Networks (OINs) are already a reality for long-distance communications [4] and their usage for short-distance communications, such as inter-chip communications, have already been proven to be applicable [5], [6]. Recently, published work presented optical 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 [7]. 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. Previous works demonstrated architectures with either long setup time (e.g., circuit switch), or too complex challenging practical deployment [8], [9]. Thus, while centralized controllers have been demonstrated for Mach-Zehnder Interferometer (MZI)-based networks [10], further improvement in their response time is required for practical deployment of OINs.

This work presents the design of a centralized controller for optical integrated networks (OINs) with a focus on a low-latency solution. The Look-up Table Centralized Controller (LUCC) enables all connections to be set-up using a clock cycle for minimum delay when the switch configuration dynamically changes. In order to do so, the LUCC is designed relying on look-up tables within a fast algorithm for conflict resolution. It takes advantage of an optimization algorithm to reduce the size of the look-up tables, and thus reduce the memory utilization.

II. Optical Integrated Networks

Optical Integrated Networks 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 two different OIN topologies: Beneš [11] and SF (Straight-Forward). While Beneš is a well known topology, SF is an in-house topology 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 [7]. Figure 1 presents the basic switches structures as well as both 8×8 topologies, which are used for validation purposes.

Manuscript received July 30, 2015; revised December 21, 2015; accepted December 26, 2015. Date of publication January 7, 2016; date of current version March 8, 2016. The associate editor coordinating the review of this paper and approving it for publication was I. B. Djordjevic.

F. G. de Magalhães is with the Computer and Software Engineering Department, Ecole Polytechnique de Montreal, Montreal, QC H3T 1J4, Canada, also with the Programa de Pós-Graduação em Ciência da Computação (PPGCC), Pontificia Universidade Catolica do Rio Grande do Sul (PUCRS), Porto Alegre 90619-900, Brazil, and also with the Department of Electrical and Computer Engineering, McGill University, Montreal, QC H3A 0G4, Canada (e-mail: felipe.magalhaes@acad.pucrs.br).

R. Priti is with Department of Electrical and Computer Engineering, McGill University, Montreal, QC H3A 0G4, Canada.

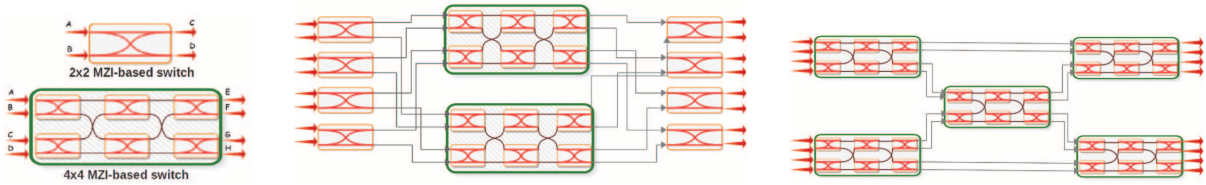
M. Nikdast is with the Computer and Software Engineering Department, Ecole Polytechnique de Montreal, Montreal, QC H3T 1J4, Canada, and also with the Department of Electrical and Computer Engineering, McGill University, Montreal, QC H3A 0G4, Canada.

F. Hessel is with the Programa de Pós-Graduação em Ciência da Computação (PPGCC), Pontificia Universidade Catolica do Rio Grande do Sul (PUCRS), Porto Alegre 90619-900, Brazil.

O. Liboiron-Ladouceur is with Department of Electrical and Computer Engineering, McGill University, Montreal, QC H3A 0G4, Canada.

G. Nicolescu is with the Computer and Software Engineering Department, Ecole Polytechnique de Montreal, Montreal, QC H3T 1J4, Canada.

Digital Object Identifier 10.1109/LCOMM.2016.2515583



(a) 2×2 and 4×4 MZI-based switches.

(b) 8×8 Beneš architecture.

(c) 8×8 SF architecture.

Fig. 1. 2 × 2 and 4 × 4 MZI-based switches and 8 × 8 MZI-based topologies schematics.

In order to describe OIN-based systems, a modelling strategy is presented, as it follows. An OIN Ψ can be defined as an n-uple $\langle n, C_i, W_i, I_i \rangle$, where:

- n: is the number of inputs and outputs in the network. For instance, an 8 × 8 network has $n = 8$;
- C_i: is a list which comprises the optical components used to route light, defined as an n-uple $\langle n_C, T_{Ci}, L_{Ci}, P_{Ci}, R_{Ci} \rangle$ and, for each C_i, the parameters stand for:
  - n_C: number of inputs and outputs. For instance, a 4 × 4 switch has $n_C = 4$;
  - T_{Ci}: is the transmission delay or the time it takes for one input to reach one output. As there may be more than one input and/or output, and the variation for each pair of I/O is small from a system-view perspective, the attributed value is the average of all I/Os;
  - L_{Ci}: stands for the insertion loss. For the same reason as above, the value used here is the average;
  - P_{Ci}: holds the power penalty of the component, and;
  - R_{Ci}: is used for the components that behave as a switch or router. It represents the routing information of the component, which maps an input to an output.
- W_i: is a list of waveguides used to connect the internal network components defined as an n-uple $\langle T_{Wi}, L_{Wi}, P_{Wi} \rangle$ and the parameters stand for transmission delay, insertion loss and power penalty[1] for each W_i, respectively, and;
- I_i: holds the I/O interfaces of the network, being defined as an n-uple $\langle T_{Ii}, L_{Ii}, P_{Ii} \rangle$ in which the parameters stand for transmission delay[1], insertion loss and power penalty for each I_i, respectively.

In addition to optical components, an electronic part is generally present on the system being responsible for the control. This controller receives requests from input nodes (processors, hardware accelerators, etc) 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 the key parameter to ensure the benefits of optics in terms of high throughput so its design must focus on reducing the impact of its execution on time.

The controller κ may be defined as an n-uple $\langle n_\kappa, S, D, A_i \rangle$, where:

- n_κ: is the number of communicating nodes using the network. For instance, in a system composed of four communicating processors, one DSP and one shared memory, $n_\kappa = 6$;
- S: defines the routing algorithm used to compute the messages path;
- D: holds the delay, in clock cycles, imposed by the processing time of the routing algorithm and network tuning, and;
- A_i: is a memory array composed by i elements used to store information related to the control, like allocation tables for dynamic network tuning.

III. Proposed Control Unit

In order to control the system, we opted for a centralized Control Unit thus decreasing the overall design complexity by providing a system's global view. As a result, LUCC's design is divided into three stages, as it follows:

1) In the first stage, the definition of a connectivity matrix[2] is made and latter a reduction method is applied [12], thus reducing control overhead by minimizing the number of possible request nodes;

2) In order to minimize the impact on the network efficiency, in the second stage, we describe the logic to compute all input requests. Learning from the iSLIP algorithm [13], we present a method to achieve the minimum overhead. Since the original iSLIP algorithm requires three cycles for computation, we modified it so all requests are treated in parallel, such that the computation is performed within one clock cycle. To do so, this block is broken into three parts: the conflict resolution, the round-robin algorithm and the control algorithm. Furthermore, as the main goal of the controller is on the low-latency, some features of the iSLIP algorithm were omitted, such as the recursive iterations. This may lead to a reduced throughput for some specific scenarios, but the gain on the control execution time alleviates this fact.

The conflict resolution works by using a matrix method, in which all I/Os are mapped to a matrix $\mathcal{M}$ of requests and then each column is checked. For instance, the matrix for a 3 × 3 switch, where all the inputs request a communication to the output three ($\langle 1 \rightarrow 3, 2 \rightarrow 3, 3 \rightarrow 3 \rangle$) is $\begin{pmatrix} 0 & 0 & 1 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \end{pmatrix}$. Using the same switch, but in a scenario such as $\langle 1 \rightarrow 3, 2 \rightarrow 1, 3 \rightarrow 2 \rangle$, the

[1]The definitions for T_{*i}, L_{*i}, P_{*i} are the same as presented for the component list.

[2]A connectivity matrix is a n × n matrix, that shows the individual connections for all nodes. For instance, the matrix $\begin{pmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{pmatrix}$ represents a scenario with n = 3 where all nodes are connected to each other but themselves.

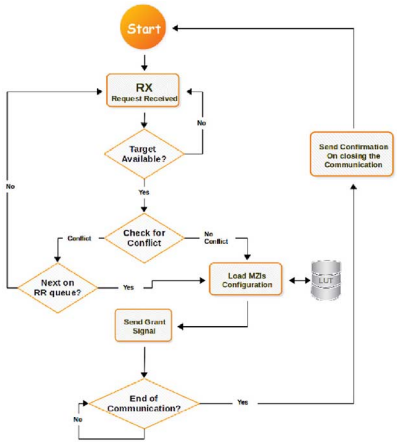


Fig. 2. Controller decision flow chart.

request matrix is $\begin{pmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix}$. The matrices are created by using the request value and requesting port position as a paired index (matrix index). For instance, using the same switch as above, the request from input one to output three creates a paired index that is equal to 1,3, which corresponds to the indexes *i,j* of the matrix. As the access to the matrix is made directly, no extra processing is needed, thus accelerating the conflict detection.

Then, all columns j of the matrix are verified for conflicting points, where a conflicting point is defined as any situation in which two or more inputs are requesting the same output. This can be defined as:

$$\forall j \in \mathcal{M}, \quad \neg XOR(j) \wedge OR(j) \implies conflict(j) = 1.$$

Following, the Round-Robin (RR) algorithm implements a first-in-first-out (FIFO) queue to decide which port will have access granted when a conflicting situation is found. As the RR uses a FIFO for each input, each request is treated individually as one particular process, which is triggered by the conflict bit controlled by the matrix method. Lastly, the control algorithm implements the intelligence of the LUCC, which processes the received requests, where each request port has its own running process.

3) Finally, in the third stage, 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 [14]. 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.

By defining these three blocks, it is possible to compute all request in one clock cycle: the conflicts' block and the RR execute independently of the clock edges, being triggered by presented signals, leaving only the control block to run constrained by the system's clock. The Figure 2 presents the decision flow chart of the controller.

IV. Low-Latency Centralized Controller

In order to validate the modelled controller, two different approaches were applied:

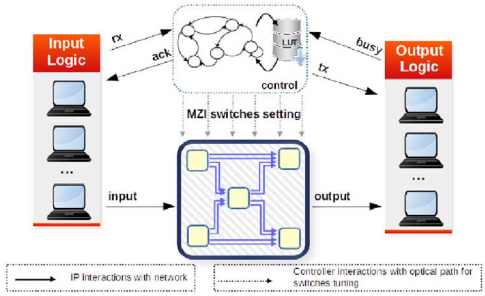


Fig. 3. An example of the validation platform.

- **Simulation**: the simulator used is an in-house platform coded in VHDL which describes all systems components using the presented models, and;
- **Prototyping**: only the LUCC is prototyped on a Stratix IV Altera's FPGA and the signal readings are performed using the Altera's SignalTap Logic Analyzer Tool [15].

A. Validation Platform

As mentioned before, the proposed controller was synthesized for the Stratix IV Altera's 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 both 8×8 topologies. Figure 3 shows an example of the validation platform, in which it is possible to see the presence of I/O nodes interacting with LUCC node and with the optical path.

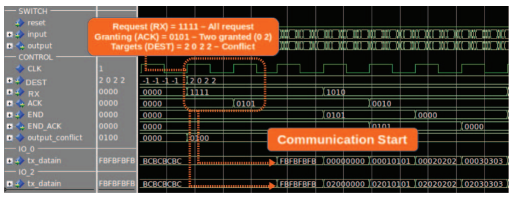
B. Results

In order to validate the LUCC, different traffic patterns, such as one-to-all and all-to-all, were applied for each topology to enable different request conditions analysis and response times. 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 for all scenarios where LUCC receives parallel requests from all inputs. In both simulation scenarios, the system is running at a frequency of 122 MHz (period $\approx$ 8 ns) and the switching delay is configured to be 40 ps. Still, it is important to highlight that the transceivers, which are injecting the traffic, run on a slightly different frequency than the system in order to correctly perform a 8 Gbps injection rate.

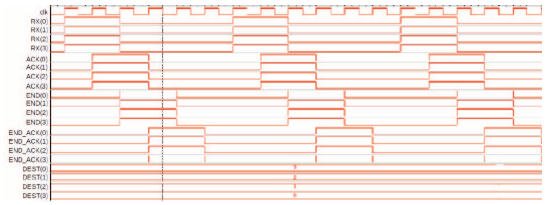
In the first scenario, simulation execution output is shown in Figure 4.(c), where four simultaneous input requests are generated (targeting (**dest** signal) outputs 0, 1, 2 and 3) and their access is granted after one clock cycle (in the image, **ack** signals are *1*, one clock cycle after **rx** signals are *1*). Still, it is possible to see the start of the traffic injection on the fast transceivers (signal **tx_datain**) and on the network simulator (signals **input** and **output**[3]).

On the second communication scenario, simulation execution output is shown on Figure 4.(a), where four

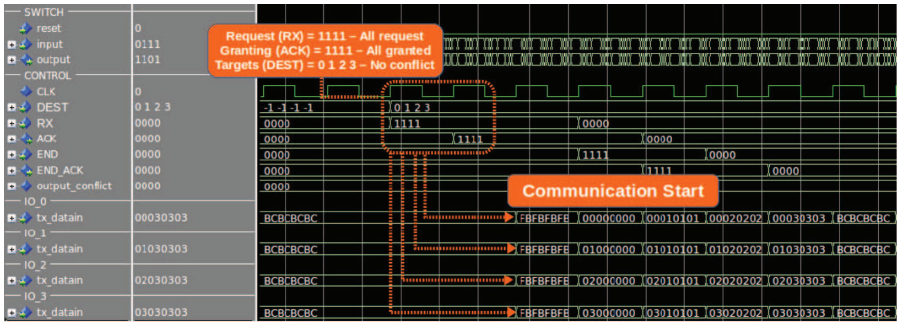
[3]The switch signals hold the payload traffic on the network. While the inserted data on the transceivers is 32 bits long, the data on the switch travels serialized, thus its waveform is different to the transceivers' waveform.



(a) Simulation of entire system with conflicts.



(b) Prototyping of LUCC showing its execution on a commercial FPGA.



(c) Simulation of entire system without conflicts.

Fig. 4. Validation results for three different scenarios: two simulations and one FPGA prototyping. For the two simulations, the system is running at 122 MHz and on the prototype, at 200 MHz. Also, the traffic injection rate from the fast transceivers is 8 Gbps. Each input is targeting one output (DEST signals).

simultaneous input requests are 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, it is still possible to see the one clock cycle granting time of the LUCC, where the traffic injection starts after the *ack* signal is sent for the granted ports (ports 0 and 2, in the Figure 4).

Finally, in the last scenario, where the LUCC prototyping can be seen on Figure 4.(b), four simultaneous input requests are generated (with no conflicts) and the on board execution response time might once again be seen from signals **rx** and **ack**. The execution frequency is 200 MHz, which gives a period of 5 ns.

V. Conclusion

This work presented the modelling, design and initial validation of LUCC, a centralized controller for Optical Integrated Networks. Results, obtained either by using an in-house high-level simulation tool for Optical Integrated Networks or by prototyping the LUCC on Altera's FPGA, showed a fast response time when employing LUCC in optical integrated networks: it takes one clock cycle delay for every request to be computed. This type of controller showed to be suitable for small to medium size topologies, by the usage of the mentioned methods of offline computing and compression. Furthermore, while this work only considers MZI-based switches, this approach can be extended to any type of switches (MZI-based or microrings) and optical integrated networks topology.

References

[1] S. Tota *et al.*, "A multiprocessor based packet-switch: Performance analysis of the communication infrastructure," in *Proc. IEEE Workshop Signal Process. Syst. Des. Implement.*, Nov. 2005, pp. 172–177.

[2] L. Benini and G. De Micheli, "Powering networks on chips: Energy-efficient and reliable interconnect design for socs," in *Proc. 14th Int. Symp. Syst. Synth. (ISSS '01)*, New York, NY, USA, 2001, pp. 33–38.

[3] International Technology Roadmap for Semiconductors (ITRS). [Online]. Available: http://www.itrs.net/, accessed on Dec. 15, 2015.

[4] A. F. Benner *et al.*, "Exploitation of optical interconnects in future server architectures," *IBM J. Res. Develop.*, vol. 49, pp. 755–775, Jul. 2005.

[5] A. V. Rylyakov *et al.*, "Silicon photonic switches hybrid-integrated with CMOS drivers," *IEEE J. Solid-State Circuits*, vol. 47, no. 1, pp. 345–354, Jan. 2012.

[6] B. G. Lee *et al.*, "Monolithic silicon integration of scaled photonic switch fabrics, CMOS logic, and device driver circuits," *J. Lightwave Technol.*, vol. 32, pp. 743–751, Feb. 2014.

[7] M. S. Hai *et al.*, "MZI-based non-blocking soi switches," in *Proc. Asia Commun. Photonics Conf.*, Paper ATh3A.147, 2014.

[8] A. Biberman *et al.*, "Broadband operation of nanophotonic router for silicon photonic networks-on-chip," *IEEE Photonics Technol. Lett.*, vol. 22, no. 12, pp. 926–928, Jun. 2010.

[9] Z. Li *et al.*, "Iris: A hybrid nanophotonic network design for high-performance and low-power on-chip communication," *J. Emerg. Technol. Comput. Syst.*, vol. 7, pp. 8:1–8:22, Jul. 2011.

[10] F. Lou *et al.*, "Towards a centralized controller for silicon photonic MZI-based interconnects," in *Proc. Opt. Interconnects Conf.—Paper WD4*, 2015, pp. 146–147.

[11] V. E. Beneš, "On rearrangeable threestage connecting networks," *Bell Syst. Tech. J.*, vol. 41, pp. 1481–1492, 1962.

[12] S. Le Beux *et al.*, "Reduction methods for adapting optical network on chip topologies to 3D architectures," *Microprocess. Microsyst.*, vol. 37, pp. 87–98, 2013.

[13] N. McKeown, "The iSLIP scheduling algorithm for input-queued switches," *IEEE/ACM Trans. Netw.*, vol. 7, no. 2, pp. 188–201, Apr. 1999.

[14] N. Jasika *et al.*, "Dijkstra's shortest path algorithm serial and parallel execution performance analysis," in *Proc. 35th Int. Conv. MIPRO*, May 2012, pp. 1811–1815.

[15] Altera. [Online]. Available: https://www.altera.com/, accessed on Dec. 15, 2015.