Low-Power Reconfigurable Network Architecture for On-Chip Photonic Interconnects

I. Artundo,* W. Heirman,† C. Debaes,* M. Loperena,* J. Van Campenhout,† H. Thienpont*

*Dept. of Applied Physics and Photonics
Vrije Universiteit Brussel, Belgium
christof.debaes@vub.ac.be

†Dept. of Electronics and Information Systems
Ghent University, Belgium
wim.heirman@ugent.be

***Abstract*—Photonic Networks-On-Chip have emerged as a viable solution for interconnecting multicore computer architectures in a power-efficient manner. Current architectures focus on large messages, however, which are not compatible with the coherence traffic found on chip multiprocessor networks. In this paper, we introduce a reconfigurable optical interconnect in which the topology is adapted automatically to the evolving traffic situation. This allows a large fraction of the (short) coherence messages to use the optical links, making our technique a better match for CMP networks when compared to existing solutions. We also evaluate the performance and power efficiency of our architecture using an assumed physical implementation based on ultra-low power optical switching devices and under realistic traffic load conditions.**

***Index Terms*—Multiprocessor interconnection, Network interfaces, Optical communication, Optical interconnections, Reconfigurable architectures, Photonic switching systems, Parallel architectures**

I. Introduction

While the number of processor cores per chip, both in Systems-on-Chip (SoCs) and Chip Multiprocessors (CMPs) keeps rising, the on-chip connections between these cores gains importance. The Network-on-Chip (NoC) paradigm is emerging as a promising solution in this space [1]. The on-chip bandwidth requirements of SoCs and CMPs are stretching the capabilities of electrical connections. Photonic communication, both inter- and intra-chip, can bring the required performance at an acceptable power usage [2], [3].

Current proposals for on-chip photonic interconnects [4], [5], [3] show compelling power and performance figures, making the case for the use of optics at these levels of the chip architecture. However, using optical links as mere drop-in replacements for the connections of electronic packet-switched networks is not the end.

Conversion at each routing point from the optical to the electrical domain and back is very power inefficient, and increases latency. Using novel components, such as silicon microring resonators [6], which can now be integrated on-chip, it is possible to build switching optical interconnection networks such as proposed by Shacham [7] or Koohi [8].

Lacking a cheap and effective way of optically controlling the routing (and doing possible buffering), these approaches necessarily work in a circuit-switched way. And while the actual switching of the optical components can nowadays be done in mere nanoseconds or less [9], the set-up of an optical circuit still requires at least one network round-trip time. This makes that these proposals only reach their full potential at large packet sizes, or in settings where software-controlled circuit switching can be used with relatively long circuit lifetimes. Indeed, in [7] packets of several kilobytes are needed to reach a point where the overhead of setting up and tearing down the optical circuits (which is done with control packets sent over an electrical network), can be amortized by the faster optical transmission. For short packets, they propose to send these directly through an electrical base network – it would make no sense to set up an optical circuit, since this is done by sending an electrical packet anyway.

In SoC architectures, and to a lesser extent in CMPs, large DMA transfers can reach packet sizes of multiple KiB. However, most packets are coherence control messages and cache line transfers. These are usually latency bound and very short. In practice, this would mean that most of the traffic would not be able to use the optical network, as they do not reach the necessary size to compensate for the latency overhead introduced, and that the promised power savings could not be realized![1]

We propose to use the combination of the electrical control network and the optical circuit-switched links as a packet-switched network with 'slow reconfiguration.' This idea is based on existing work such as the Interconnection Cached Network [10] (or see [11] for a modern application). But rather than relying on application control of the network reconfiguration, which requires explicit software intervention and does not agree with the implicit communication paradigm of the shared memory programming model, our approach provides for an automatic reconfiguration based on the current network traffic. This concept has been described in [12], and was proven to provide significant performance benefits in (off-chip) multiprocessor settings. In this paper, we will extend this approach to on-chip networks, trying to model an architecture close to the one already introduced in [7].

II. On-chip photonic interconnect architecture

The photonic NoC proposed by Petracca et al. [13] introduces a non-blocking mesh topology, connecting the different

I. Artundo is now with the Optical Communications Group at iTEAM-UPV (Valencia, SPAIN).

[1]One might consider using a larger cache line size to counter this, but an increase to multiple KiB would in most cases only result in excessive amounts of *false sharing*, negating any obtained performance increase.

1550-4794/09 $26.00 © 2009 IEEE
DOI 10.1109/HOTI.2009.27

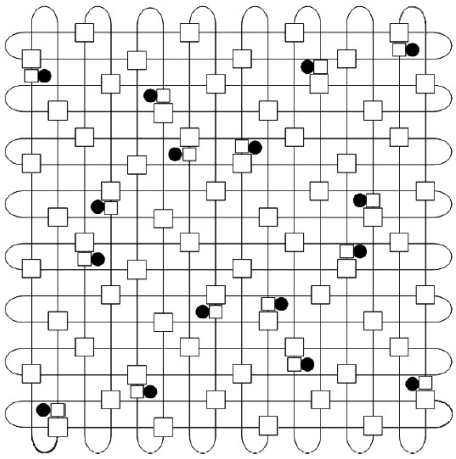

Fig. 1. 16-node non-blocking torus [13]. Squares represent optical routers based on microring resonators, the network nodes are represented by discs. The electrical control (or base) network, which is a 2-D torus overlayed on the optical network, is not shown here.

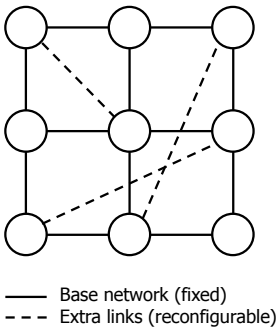

Fig. 2. Reconfigurable network topology. The network consists of a base network (a 2-D torus in our architecture), augmented with a limited number of direct, reconfigurable links (which are made up of the reconfigurable optical layer from Figure 1).

cores of the system, based on a hybrid approach: a high-bandwidth circuit-switched photonic network is combined with a low-bandwidth packet-switched electronic network. This way, large data packets are routed through a time and wavelength multiplexed network, for a combined bandwidth of 960 Gbps, while delay-critical control packets and some smaller data messages are routed through the low-latency electrical layer. As the basic switching element, a 4×4 hitless silicon router is presented in [14], based on eight silicon microring resonators with a bandwidth per port of 38.5 GHz on a single wavelength configuration.

An example 16-node architecture is depicted in Fig. 1. Each square represents a 4×4 router containing eight microring resonators. The smaller squares are the inject/eject 3×3 routers. The network nodes themselves are represented by discs.

In this architecture, each node has a dedicated router to inject and eject packets from the network. By means of the electronic control layer, each node first sends a control packet to make the reservation of a photonic circuit from source to destination. Once this is done, transmission is done uninterrupted for all data packets. To end the transmission phase, a control packet is sent back from destination to free the allocated resources.

For our architecture, a dedicated reconfigurable photonic layer will be used as a data transmission layer, where a set of extra links will be established in a circuit-switched fashion for certain intervals of time, depending on automated load measurements over the base topology. The reconfiguration will follow slow-changing dynamics of the traffic and the base electronic network layer will still be there to route control and data messages too.

Other architectures have been proposed, such as [15], where the need for an electrical control layer has been removed, and all packets are sent through an all-optical network using different wavelengths. Still, the separation between control and data layers, even when they are sent through the same physical channels, is maintained. Our approach is valid to any network architecture where this distinction is kept, as the reconfigurable layer can be virtually established irrespective of the underlying physical implementation.

III. Reconfigurable optical layer

A. Using traffic locality to trigger reconfiguration

It is known that memory references exhibit locality in space and time, in a fractal or self-similar way. This locality is exploited by caches. Due to the self-similar nature of locality, this effect is present at all time scales, from the very fast nanosecond scales exploited by first-level caches, down to micro- and millisecond scales which are visible on the interconnection network of a shared-memory (on-chip or multi-chip) multiprocessor. This behavior can be modeled as traffic bursts: these are periods of high-intensity communication between specific processor pairs. These bursts were observed to be active for up to several milliseconds, on a background of more uniform traffic with a much lower intensity.

From this observation came the idea to use slowly reconfigurable but high (data-) speed optical components to establish 'extra links,' providing direct connections between pairs of processor cores that are involved in a communication burst. Other communication, which is not part of a burst – or a lower-intensity burst when the hardware would support less extra links than there are bursts at a given time – will be routed through a standard packet switched (optical or electrical) network (the 'base network,' see Figure 2). The positions of the extra links are re-evaluated over time as old bursts stop and new ones appear.

We have evaluated this concept in the context of shared-memory servers and supercomputers, and proposed an implementation using low-cost optical components in [12]. Since then, multicore technology has enabled the integration of a complete shared-memory multiprocessor on a single chip. At the same time, on-chip reconfigurable optical interconnects became a reality, using the integration possibilities allowed by the emerging field of silicon photonics [16].

B. Proposed reconfigurable network architecture

Our network architecture, originally proposed in [12], starts from a base network with fixed topology. In addition, we provide a second network that can realize a limited number of connections between arbitrary node pairs – the *extra links* or *elinks*. A schematic overview is given in Figure 2.

The elinks are placed such that most of the traffic has a short path (a low number of intermediate nodes) between source and destination. This way a large percentage of packets has a correspondingly low (uncongested) latency. In addition, congestion is lowered because heavy traffic is no longer spread out over a large number of intermediate links. For the allocation of the elinks, a heuristic is used that tries to minimize the aggregate hop distance traveled multiplied by the size of each packet sent over the network, under a set of implementation-specific conditions: these can be the maximum number of elinks n, the number of elinks that can terminate at one node (the *fanout*, f), etc. After each interval of length Δt (the reconfiguration interval), a new optimum topology is computed using the traffic pattern measured in a *previous* interval.

This process requires some of the collected results to be exchanged over the network and includes the optimization algorithm itself. It therefore cannot be assumed negligible. The time this exchange and calculation takes will be denoted by the the *selection time* (t_{Se}). The actual switching of optical reconfigurable components will then take place during a certain *switching time* (t_{Sw}), after which the new set of elinks will be operational. Traffic cannot be flowing through the elinks while they are being reconfigured. Therefore, the reconfiguration process starts by draining all elinks before switching any of the microrings. This takes at most 20 ns (the time to send our largest packet, which is 80 bytes, over a 40 Gbps link). During the whole reconfiguration phase, network packets can still use the base network, this makes our technique much less costly than some other, more intrusive reconfiguration schemes, where all network traffic needs to be stopped, and drained from the complete network, during reconfiguration.

The reconfiguration interval Δt must be chosen as short as possible to be able to follow the dynamics of the evolving traffic and get a close-to-optimal topology. On the other hand, it must be significantly larger than the switching time of the chosen implementation technology to amortize the fraction of time that the *elinks* are off-line.

Gathering traffic information for each of the nodes to compute the optimal network configuration is straightforward if each node can count the number of bytes sent to each destination. Collecting this data at a centralized arbiter over our high-performance interconnect only takes one network round-trip time. Finally, computation needs to be done on this data at the centralized unit. This computation is largely based on heuristics and pre-computed tables, and can therefore quickly determine a near-optimal elink configuration and its corresponding routing tables. We assume that this selection algorithm can be executed on one of the system's UltraSPARC processors, and even for a 64-node network we expect this to take only a few microseconds. Of course, this will only hold for slowly-reconfiguring networks, where the reconfiguration interval is long enough to amortize this delay.

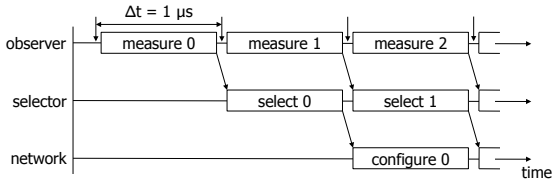

Fig. 3. Sequence of events in the on-chip reconfigurable network. During every reconfiguration interval of 1 µs, a traffic pattern is measured. In the next interval, the optimal network configuration is computed for this traffic pattern. One interval later, this configuration is enabled. The reconfiguration itself takes place at the start of each *configure* box, but the switching time is very short in this architecture and is therefore not shown here.

If we want to reduce the reconfiguration interval even further, we will have to move to a decentralized scheme, where traffic information is spread locally to neighboring nodes only, and the selection mechanism is done at each processor with just local information.

C. Mapping the reconfigurable architecture onto the photonic network

Applying this architecture to the specifics of a NoC, we can consider the network presented in [13] as equivalent to such a reconfigurable network, where the number of elinks n equals the number of processing nodes p, and with a maximum fan-out per node of one ($n = p$, $f = 1$). This way, each extra link would be considered as a dedicated circuit of the non-blocking mesh, and therefore, our existing simulation modules could be reused for a performance evaluation. The reconfiguration interval, Δt, was fixed in this case to 1 µs.

With optical components that can switch in the 30 ps range, the switching time (t_{Sw}) will only take a negligible fraction of the reconfiguration interval Δt. However the selection time (t_{Se}) will remain significant as it requires exchange of data over the network.

We therefore propose a scheduling where we allow the selection to take up to a full reconfiguration interval. This pushes the actual reconfiguration into the next interval. The three activities (shown in Figure 3) of collecting traffic information (*measure*), making a new elink selection (*select*), and adjusting the network with this selection (*configure*) are performed in pipelined fashion, where each activity uses the results (traffic pattern or elink selection) from one interval ago.

IV. Methodology

A. Simulation platform

We have based our simulation platform on the commercially available Simics simulator [17]. It was configured to simulate a chip multiprocessor with 16 or 64 UltraSPARC III processor cores clocked at 2.5 GHz and running the Solaris 10 operating system. Each core was considered to be one network node. For a second experiment, we used the multicore UltraSPARC T1/T2 processor, which runs multiple (four in our

case) threads per core. This way, the traffic of 64 threads was concentrated on a 16-node network, stressing the interconnection network with aggregated traffic. Stall times for caches and main memory are set to realistic values (2 cycles access time for L1 caches, 19 cycles for L2 and 100 cycles for main memory). Cache coherence is maintained by a directory-based coherence controller at each node, which uses a full bit vector directory and an MSI-protocol. Since all writes to a specific cache line serialize at its home node directory, this guarantees (at least release) consistency. By making a few small changes to the standard coherence protocol we could tolerate out-of-order delivery by the reconfigurable network, obviating the need for reorder buffers. The interconnection network models a packet-switched 4×4 or 8×8 torus network with contention and cut-through routing. The time required for a packet to traverse a router is 3 cycles. Both the directory controller and the interconnection network are custom extensions to Simics.

The network links in the base network have a speed of 10 Gbps. To model the elinks, a number of extra point-to-point links can be added to the torus topology at any point in the simulation. The speed of the optical elinks were assumed to be eight times faster (40 Gbps). Both coherence traffic (read requests, invalidation messages etc.) and data are sent over the base network. The resulting remote memory access times are around 100 ns, depending on network size and congestion.

To avoid deadlocks, dimension order routing is used on the base network. Each packet can go through one elink on its path, after that it switches to another virtual channel (VC)[2] to avoid deadlocks of packets across elinks. For routing packets through the elinks we use a static routing table: when reconfiguring the network, the routing table in each node is updated such that for each destination it tells the node to route packets either through an elink starting at that node, to the start of an elink on another node, or straight to its destination, the latter two using normal dimension order routing.

The network traffic is the result of both coherence misses and cold/capacity/conflict misses. To make sure that private data transfer does not become excessive, a first-touch memory allocation was used that places data pages of 8 KB on the node of the processor core that first references them. Also each thread is pinned down to one processor (using the Solaris `processor_bind()` system call), so the thread stays on the same network node as its private data for the duration of the program.

The SPLASH-2 benchmark suite [18] was chosen as the workload. It consists of a number of scientific and technical algorithms using a multi-threaded, shared-memory programming model (`barnes`, `cholesky`, `fft`, `radix`, `ocean.cont`, `water.sp`). Because the default benchmark sizes are too big to simulate their execution in a reasonable time, smaller problem sizes were used. Since this affects the working set, and thus the cache hit rate, the level 2 cache was resized from an actual 8 MiB on a real UltraSPARC III to 512 KiB. Also the associativity was increased to 4-way (compared to 2-way for the US-III) after we experienced excessive conflict misses in Solaris' internal structures with the 2-way caches. Overall, this resulted in realistic 93–97% hit rates for the L2 caches. 50–60% of L2 misses were cataloged as coherence misses (resulting in communication between different processors), the remaining 40–50% were cold/conflict/capacity misses.

Since the detailed simulation of a single SPLASH-2 benchmark program takes a significant amount of computation time, especially for the larger networks, we used synthetic traffic traces instead. For each of the benchmark applications and network sizes an individual trace is constructed using the methodology introduced in [19]. This way, we could quickly yet accurately simulate the performance and power consumption of our network under realistic traffic conditions.

B. Extra link selection

For every reconfiguration interval, a decision has to be made on which elinks to activate, within the constraints imposed by the architecture, and based on the expected traffic during that interval. In our current implementation, the traffic is expected to be equal to the traffic that was measured two intervals ago – this avoids the need for a complicated and time-consuming prediction algorithm. As explained in Section III-B, we want to minimize the number of hops on the (electronic) base network for most of the traffic. We do this by minimizing a cost function that expresses the total number of network hops traversed by all bytes being transferred. This cost function can be written as:

$$C = \sum_{i,j} d(i,j) \cdot T(i,j) \tag{1}$$

with $d(i,j)$ the distance between nodes i and j, which is a function of the elinks that are selected to be active, and $T(i,j)$ the number of bytes sent from node i to node j in the time interval of interest.

Since the time available to perform this optimization is equal to the reconfiguration time (1 µs here), we use a greedy heuristic that can quickly find a set of active elinks that satisfies the constraints imposed by the architecture, and has an associated cost close to the global optimum. More details on this algorithm can be found in [20].

C. Power measurements

To measure the power consumption of our optical circuit-switched routing, we will need to know the state of each switch in the mesh – this means which microrings are powered on for each reconfiguration interval. We can know this by looking at the routing table of each router (see [14], Table 1b) and assigning a power number for each active ring. In [14], the power consumed per ring in the ON state is assumed to be 6.5 mW, in the OFF state the required power is considered negligible. This is for rings that switch in only 30 ps, though. Using a reconfiguration interval of 1 µs, our architecture doesn't need such an exorbitantly fast (and power hungry) device. Instead, it can tolerate several nanoseconds of switching time, and we will assume that such a device can be powered with just 0.4 mW.

In [14], nine possible states of the router are considered, determined by all possible simultaneous connections between

[2]Actually another *set* of VCs is used since we already employ separate request and reply VCs to avoid fetch deadlocks at the protocol level.

Technology Node	32 nm
Core Dimension	1.67×1.67 mm^2
Electrical Link Power	0.34 pJ/bit/mm
Static Electrical Link Power @ 10 Gbps	500 µW
Static Electrical Link Power @ 40 Gbps	2 mW
Optical Link Power	0.5 pJ/bit
Static Optical Link Power	500 µW
Microring ON Power	400 µW
Microring OFF Power	0 µW
Buffering Energy	0.12 pJ/bit
Crossbar Transfer Energy	0.36 pJ/bit
Static Routing Energy	0.35 pJ/bit

TABLE I
POWER CONSUMPTION FIGURES.

its in- and outputs. Each of these states has a specific number of microrings powered on. However, when a router is only used by a single traversing elink, fewer active microrings are required. If we do not consider the nine predefined states, but only account for the minimal number of rings needed for establishing the optical elink path, we can obtain a significantly lower power consumption.

Therefore, one could think of a more power-efficient scheme that only powers the rings needed on each reconfiguration interval, instead of putting the switch in a state where several rings will be powered whether they are used or not. Of course, the electronic control of such a switch would be more complicated, this is why the nine predefined states are proposed in [14], even if this is not the most power-efficient scheme. But where the localized control, and the aim for independence between the different circuits considered in [14] validates such an approach, our architecture on the other hand performs a global and simultaneous assignment of all elinks and microrings and should therefore be able to operate in the optimized case.

For the parameters to estimate the power consumption of links and the routing of the packets, we have used the same values as cited in [7] and shown in Table I. One notable difference is that we include an extra static power of 500 µW for each optical link, as it is likely that the analog optical transceiver circuits will consume power even while the links are not sending data. As for the dynamic power dissipated by the E/O and O/E conversion, a reasonable estimate for a modulator and its corresponding detector at 10 Gbps is 2 pJ/bit. Future predictions push this value down to 0.2 pJ/bit [21].

V. SIMULATION RESULTS

A direct comparison with our reference architecture of [14] is difficult, since in the original case only large DMA transfers (of which there can be very few in realistic CMP systems) would use the optical network, while most of the traffic – both by aggregate size and by latency sensitivity – necessarily sticks to the electrical 'control' network. Yet, just comparing the performance of our solution with a base-network only is not very insightful either. Therefore, we will make a performance and power comparison of our proposed architecture with an all-electrical, and also with an all-optical solution, which are both implemented as a non-reconfigurable 2-D torus topology.

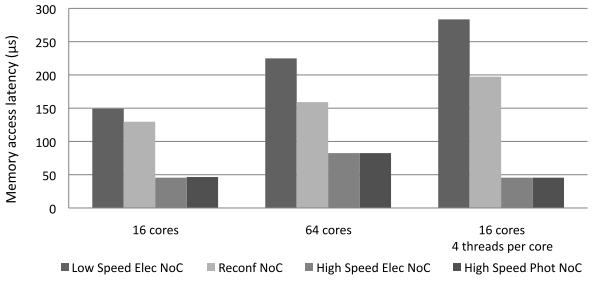

Fig. 4. Average remote memory access latency.

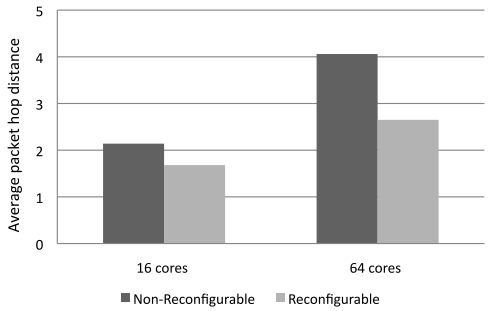

Fig. 5. Average number of hops per byte sent.

A. Network performance

In this section we first aim to obtain the performance improvement by introducing reconfiguration in the system, versus a standard topology. For this, we compare four approaches: using either the reconfigurable architecture introduced above, or a 2-D torus-only network with link speeds of 10 Gbps ('Low speed electrical NoC') or 40 Gbps ('high speed') electrical or optical NoC, without reconfiguration capabilities. In the case of an all-optical network, every node needs an optical transceiver in all four directions. Also, a conversion from the optical to the electrical domain is needed at each hop, since the routing is still performed electronically. On this other hand, in the case of the reconfigure network we require only one transceiver per node, which is an advantage in cost and power consumption. Moreover, the data can now travel over much longer distances until O/E and E/O conversions are needed, which again reduces power and latency.

In Figure 4, average remote memory access latencies are presented for all network configurations. We can observe that the reconfigurable approach performs significantly better than the low-speed non-reconfigurable network, but still far from a high-speed electrical/optical implementation due to the huge amount of bandwidth available in this case. This already gives the hint that bandwidth will be the key factor here.

B. Power consumption

Now we will evaluate the power used by powering the microring resonators when establishing the elinks on the reconfigurable layer.

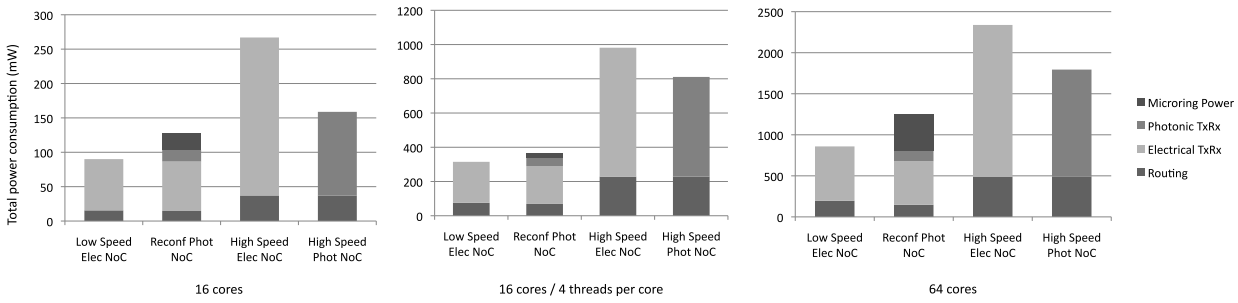

Fig. 6. Total power consumption per interval under different network architectures.

In Figure 5 we show the average number of hops per byte sent, which will help us compute the consumed power in transmission by assigning power figures to each network case as described before in Section IV-C. Comparing with the non-reconfigurable topology, in which the network consists of just a 2-D torus, there is a clear reduction of the hop distance by 21.5% to 34.7% as we move to larger sized networks.

If we consider then a network of $p = 16$ processing nodes, for the case of switches on nine predetermined states, the mean power consumed by the active microring resonators per reconfiguration interval is 45.2 mW (counting all active microring resonators in all switches), when powering only the minimal number of rings in each switch only 29.5 mW is required, this is a reduction by 65%.

With a network of $p = 64$ processing nodes, in the case of considering finite-state switches, the mean power consumed by the active microring resonators in the switches, per interval, is 720 mW. For the minimal case, this value is decreased to 425 mW, a reduction by 60%.

There is only a small variability between the different applications measured because, at any time, there is exactly the same number of elinks present. The only thing that can differ is that, sometimes, slightly longer routes are created, but since the elink selection always tries to maximize *data* $\times$ *hop-distance*, the average hop distance will also be not that different. Note that the number of active microrings depends on the shape of the traffic pattern (the source-destination pair distribution) – albeit not by a great amount – but that it does not depend on the traffic magnitude. The power computed above for the 16-node case is therefor valid both for the simple 16-processor case as for the multithreaded situation.

Finally, in Figure 6 we present a summary of the power consumption for all network architectures considered, according to the measurements explained in Section IV-C. As can be seen, the reconfigurable network consumes more than a slow 10 Gbps non-reconfigurable electrical network, but much less than its electrical or optical counterparts at 40 Gbps. Again, as we increase the size of the network, differences become more evident. Traffic aggregation on the UltraSPARC T2 case makes reconfiguration become more profitable, as traffic on the base network increases significantly and both electrical and optical high-speed networks experience higher activity. The limiting factor on the reconfigurable network will be the power consumed by the active microrings while scaling up.

	16 nodes	**64 nodes**
Total number of switches in the network	64	1024
Number of switches used per elink	3.28	15.3
Total number of switches used	37.5	692
Switches re-used by multiple elinks	13	251
Average number of active microrings	74	1073

TABLE II
AVERAGE SWITCH USAGE PER RECONFIGURATION INTERVAL.

C. Network usage

A key factor in understanding the power consumption is the usage of the switches and links in the network. For a normal $r \times \frac{p}{r}$ torus topology, the diameter (maximum number of hops between any node pair) is [22]:

$$D = \left\lfloor \frac{r}{2} \right\rfloor + \left\lfloor \frac{p}{2r} \right\rfloor \quad (2)$$

where p is the number of processors and r is the size of the torus. In regular tori this makes $D = 4$ hops for $p = 16$, and $D = 8$ for $p = 64$. The average hop distance is 2.13 for $p = 16$, and 4.06 for $p = 64$.

In our simulations, we use a folded torus topology as shown in Figure 1. The complete topology contains $\frac{p^2}{4}$ switches (4×4 routers) and p gateway switches. We found that the mean number of (non-gateway) switches used per elink during each reconfiguration interval is 3.28 for the $p = 16$ case. This results in a total of 37.5 active routers (out of the 64 available ones), of which 13 routers are traversed by more than one elink. Inside all routers, on average 73.7 microrings are in the active state. These numbers, and those for the 64-node network, can be found in Table II.

The folded torus topology used in our study has twice the wire demand and bisection bandwidth of a mesh network, trading a longer average flit transmission distance for fewer routing hops. While wider flits and a folded topology can increase link bandwidth utilization efficiency, this remains still low in our simulations, as shown in Table III.

Pande et al. [23] investigated various metrics of a folded torus NoC, including energy dissipation, for different traffic loads. The comparative analysis was done with respect to average dynamic energy dissipated per full packet transfer from source to destination node. It was found that energy dissipation increases linearly with the number of VCs used. Furthermore,

	16 nodes	64 nodes
Reconfigurable	27.1%	37.7%
All-electrical	23.4%	40.4%
All-optical	6.8%	30.5%

TABLE III
NETWORK LINK ACTIVITY RATIOS.

a small number of VCs will keep energy dissipation low without giving up throughput. Energy dissipation reaches an upper limit when throughput is maximized, meaning that energy dissipation does not increase beyond the link saturation point. In general, architectures with more elaborate topologies, and therefore higher degrees of connectivity, have a higher energy dissipation on average, at this saturation point, than do others. If, as it is always the case in CMP NoC, power dissipation is critical, a simpler mesh topology may be preferable to a folded torus, as detailed in [1].

VI. CONCLUSIONS

We introduced a reconfigurable optical interconnect for a NoC multicore system that makes use of ultra-low power photonic switches to route messages over a reconfigurable optical layer, while keeping an underlaying electronic base network. Since we allow for slow reconfiguration, or adaptation of the optical layer to the current traffic pattern, our approach can make much better use of the optical layer – which otherwise would only be beneficial for very long packets, or for circuits that were explicitly set up by the programmer. Both these conditions are however not compatible with realistic chip multiprocessor architectures. By using our approach, however, the full benefits of optical switching can be combined with realistic CMP conditions, paving the way for photonic interconnects to satisfy the future bandwidth needs of large multicore designs.

ACKNOWLEDGMENTS

This work was supported by the European Commission's 6th FP Network of Excellence on Micro-Optics (NEMO), the IAP P6/10 photonics@be network sponsored by the Belgian Science Policy Office, the FWO, the OZR, the Methusalem and Hercules foundations.

REFERENCES

[1] W. J. Dally and B. Towles, "Route packets, not wires: On-chip interconnection networks," in *Design Automation Conference*, Jun. 2001, pp. 684–689.

[2] J. D. Owens, W. J. Dally, R. Ho, D. Jayasimha, S. W. Keckler, and L.-S. Peh, "Research challenges for on-chip interconnection networks," *IEEE Micro*, vol. 27, no. 5, pp. 96–108, 2007.

[3] K. Ohashi, K. Nishi, T. Shimizu, M. Nakada, J. Fujikata, J. Ushida, S. Torii, K. Nose, M. Mizuno, H. Yukawa, M. Kinoshita, N. Suzuki, A. Gomyo, T. Ishi, D. Okamoto, K. Furue, T. Ueno, T. Tsuchizawa, T. Watanabe, K. Yamada, S. Itabashi, and J. Akedo, "On-chip optical interconnect," *Proceedings of the IEEE*, vol. 97, no. 7, pp. 1186–1198, Jul. 2009.

[4] M. Brière, B. Girodias, Y. Bouchebaba, G. Nicolescu, F. Mieyeville, F. Gaffiot, and I. O'Connor, "System level assessment of an optical NoC in an MPSoC platform," in *Design, Automation and Test in Europe (DATE)*, 2007, pp. 1084–1089.

[5] R. G. Beausoleil, J. Ahn, N. Binkert, A. Davis, D. Fattal, M. Fiorentino, N. P. Jouppi, M. McLaren, C. M. Santori, R. S. Schreiber, S. M. Spillane, D. Vantrease, and Q. Xu., "A nanophotonic interconnect for high-performance many-core computation," *IEEE LEOS Newsletter*, pp. 15–22, Jun. 2008.

[6] Q. Xu, D. Fattal, and R. G. Beausoleil, "Silicon microring resonators with 1.5-μm radius," *Optics Express*, vol. 16, no. 6, p. 4309, 2008.

[7] A. Shacham, K. Bergman, and L. Carloni, "Photonic networks-on-chip for future generations of chip multi-processors," *IEEE Transactions on Computers*, vol. 57, no. 9, pp. 1246–1260, Sep. 2008.

[8] S. Koohi and S. Hessabi, "Contention-free on-chip routing of optical packets," in *Proceedings of the 3rd ACM/IEEE International Symposium on Networks-on-Chip*, May 2009, pp. 134–143.

[9] O. Fidaner, H. V. Demir, V. A. Sabnis, J.-F. Zheng, J. S. J. Harris, and D. A. B. Miller, "Integrated photonic switches for nanosecond packet-switched optical wavelength conversion," *Optics Express*, vol. 14, no. 1, p. 361, 2006.

[10] V. Gupta and E. Schenfeld, "Performance analysis of a synchronous, circuit-switched interconnection cached network," in *ICS '94: Proceedings of the 8th international conference on Supercomputing*. Manchester, England: ACM, Jul. 1994, pp. 246–255.

[11] K. J. Barker, A. Benner, R. Hoare, A. Hoisie, A. K. Jones, D. K. Kerbyson, D. Li, R. Melhem, R. Rajamony, E. Schenfeld, S. Shao, C. Stunkel, and P. Walker, "On the feasibility of optical circuit switching for high performance computing systems," in *SC '05: Proceedings of the 2005 ACM/IEEE conference on Supercomputing*. Washington, DC: IEEE Computer Society, Nov. 2005, p. 16.

[12] I. Artundo, L. Desmet, W. Heirman, C. Debaes, J. Dambre, J. Van Campenhout, and H. Thienpont, "Selective optical broadcast component for reconfigurable multiprocessor interconnects," *IEEE Journal of Selected Topics in Quantum Electronics: Special Issue on Optical Communication*, vol. 12, no. 4, pp. 828–837, Jul. 2006.

[13] M. Petracca, B. G. Lee, K. Bergman, and L. P. Carloni, "Design exploration of optical interconnection networks for chip multiprocessors," in *Proceedings of the 16th IEEE Symposium on High Performance Interconnects*, Stanford, California, Aug. 2008, pp. 31–40.

[14] N. Sherwood-Droz, H. Wang, L. Chen, B. G. Lee, A. Biberman, K. Bergman, and M. Lipson, "Optical 4×4 hitless slicon router for optical networks-on-chip (NoC)," *Optics Express*, vol. 16, no. 20, pp. 15 915–15 922, 2008.

[15] H. Gu, J. Xu, and W. Zhang, "A low-power fat tree-based optical network-on-chip for multiprocessor system-on-chip," in *Design, Automation and Test in Europe (DATE)*, Nice, France, Apr. 2009, pp. 3–8.

[16] Y. Vlasov, W. M. J. Green, and F. Xia, "High-throughput silicon nanophotonic wavelength-insensitive switch for on-chip optical networks," *Nature Photonics*, vol. 2, pp. 242–246, 2008.

[17] P. S. Magnusson, M. Christensson, J. Eskilson, D. Forsgren, G. Hallberg, J. Hogberg, F. Larsson, A. Moestedt, and B. Werner, "Simics: A full system simulation platform," *IEEE Computer*, vol. 35, no. 2, pp. 50–58, Feb. 2002.

[18] S. C. Woo, M. Ohara, E. Torrie, J. P. Singh, and A. Gupta, "The SPLASH-2 programs: Characterization and methodological considerations," in *Proceedings of the 22th International Symposium on Computer Architecture*, Santa Margherita Ligure, Italy, Jun. 1995, pp. 24–36.

[19] W. Heirman, J. Dambre, and J. Van Campenhout, "Synthetic traffic generation as a tool for dynamic interconnect evaluation," in *Proceedings of the 2007 International Workshop on System Level Interconnect Prediction (SLIP'07)*. Austin, Texas: ACM Press, Mar. 2007, pp. 65–72.

[20] W. Heirman, J. Dambre, I. Artundo, C. Debaes, H. Thienpont, D. Stroobandt, and J. Van Campenhout, "Predicting the performance of reconfigurable optical interconnects in distributed shared-memory systems," *Photonic Network Communications*, vol. 15, no. 1, pp. 25–40, Feb. 2008.

[21] W. M. J. Green, M. J. Rooks, L. Sekaric, and Y. A. Vlasov, "Ultra-compact, low RF power, 10 Gb/s silicon mach-zehnder modulator," *Optics Express*, vol. 15, no. 25, pp. 17 106–17 113, 2007.

[22] B. Parhami, *Introduction to Parallel Processing: Algorithms and Architectures*. Kluwer Academic Publishers, 1999.

[23] P. P. Pande, C. Grecu, M. Jones, A. Ivanov, and R. Saleh, "Performance evaluation and design trade-offs for network-on-chip interconnect architectures," *IEEE Transactions on Computers*, vol. 54, no. 8, pp. 1025–1040, Aug. 2005.