Coherence Based Message Prediction for Optically Interconnected Chip Multiprocessors

Anouk Van Laer, Chamath Ellawala
Muhammad Ridwan Madarbux, Philip M. Watts
Dept. of Electronic and Electrical Engineering, University College London
London, United Kingdom
Email: anouk.vanlaer,chamath.ellawala,m.madarbux,philip.watts@ucl.ac.uk

Timothy M. Jones
Computer Laboratory
University of Cambridge
Cambridge, United Kingdom
Email: timothy.jones@cl.cam.ac.uk

***Abstract*—Photonic networks on chip have been proposed to reduce latency and power consumption of on-chip communication in chip multiprocessors. However, in switched photonic networks, the path setup latency can create a high overhead, particularly for the short messages generated by shared memory chip multiprocessors (CMP). This has led to proposals for networks which avoid switching using all-to-all or single writer multiple reader (SWMR) networks which dramatically increase optical component counts and hence power consumption. In this work we propose a predictor which uses information from the coherence protocol and previously transmitted messages to predict future messages and hence hide the path setup latency by speculatively setup photonic paths. We show that a directly mapped predictor can achieve prediction hit rates of up to 85% for PARSEC benchmarks in a 16-core x86 system using the MESI coherence protocol whereas a more resource efficient set associative predictor can still achieve prediction rates up to 75%.**

I. Introduction

Due to the increasing number of cores per chip, networks-on-chip (NoC) have replaced bus based communication in chip multiprocessors (CMPs). Electronic NoCs in existing CMPs consist of meshes [1] or pipelined crossbars [2] where messages travel the network hop-by-hop between electrical buffers. Photonic NoCs have been proposed to reduce the power consumption and ease thermal management issues of CMPs [3]–[7]. The absence of viable optical buffers requires end-to-end paths being setup before the actual communication can start, in contrast to the hop-by-hop approach used in the electronic NoCs. Contributions to the total message latency in a scheduled photonic switched network are shown in Figure 1 consisting of path setup latency, time of flight in the waveguide and serialization latency. The serialization latency can be reduced by increasing bit rate or modulating and switching the message in parallel on multiple wavelengths (wavelength striping) [3]. However, the overhead of the path setup phase can be significant, especially in shared memory CMPs where the majority of the messages are short coherence control messages. This has led to many photonic NoC proposals which avoid the path setup overhead by providing constant photonic paths between all possible source-destination pairs [5], [6]. The disadvantage of these non-switching NoCs lies in their high photonic component count, low bandwidth per node and the resulting high serialisation latency. In this work, we propose the use of a wavelength striped photonic switch with a coherence protocol-based predictor to start the path setup before the actual messages arrive at the network ingress, in effect hiding the path setup latency. In contrast to previous work [8], the predictor described here can also accelerate complex transactions involving three or more cores or main memory rather than purely simple request-response transactions.

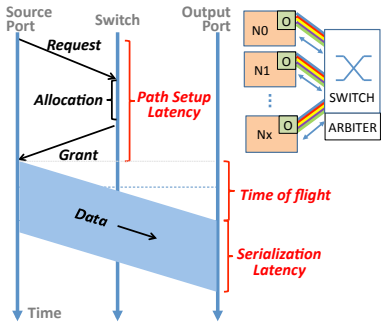

Fig. 1. Sources of latency in a scheduled photonic switch.

II. Background

Various techniques have been proposed to avoid the path setup latency of photonic networks. Firstly, arbitration can be avoided by using a non-switched network. In [5] all-to-all connectivity is provided by a combination of space and wavelength routing. In the single writer, multiple reader scheme [6], every source gets dedicated wavelength to write to. While both proposals avoid arbitration, they have increased power consumption and optical component count which are further increased if additional wavelengths are used to reduce serialisation latency. Another possibility is the use of speculative transmission [3] in which messages are transmitted without securing a path beforehand. While this reduces the latency associated with arbitration, scheduling is still on the critical path and the network needs to handle dropped or reordered messages [9]. Various prediction schemes have been proposed to reduce the latency of memory requests in electronic NoCs. In [10], the need for cache-to-cache transfers is predicted based by combining the program counter and memory addresses.

In [11] prediction is used to forward memory addresses to future readers, thus avoiding L1 misses and the associated messages to the directory. In [12] a cache coherence protocol is proposed in which other caches sharing a cacheline are predicted to avoid indirection via the directory. While these proposals decrease the latency of memory requests by avoiding unnecessary network transactions, they do not speedup the messages that still need to travel the NoC. In [13] prediction is used to reduce the setup latency of a hybrid optical circuit/electrical mesh network by using channel prediction in the electrical routers in combination with lookahead routing. In contrast, this paper proposes the use of a relatively simple, low power photonic switched NoC combined with a coherence based predictor which sets up optical paths for most messages in advance.

III. Predictor setup

Messages in shared memory CMP are not randomly injected into the network, they are part of a coherence transaction. Examples of such transactions are given in Figure 2. In the rest of the paper, messages are described by their source, type and size (control/data). A *L1 REQ C* message for example is a short (8B) request message leaving a L1-cache controller. This type of message signifies the start of a transaction. Both Figure 2(a) and (b) show the coherence transactions in the MESI protocol following a ST access to address *a*. In Figure 2(a) the address is already present in L1 and needs to be upgraded to the exclusive state but other L1 caches also have a copy of *a*. All other sharers need to acknowledge their invalidation to L1, before the upgrade can complete. The length of the upgrade transaction depends on the number of sharers. In Figure 2(b) the address is not yet present in the on-chip cache hierarchy and needs to be retrieved from main memory.

The previous examples can be generalized: when the CPU makes a request for an address, the resulting coherence transaction and its associated messages seen by the network are a combination of the type of access requested and the current state the requested address is in. For example, in the case of the upgrade transaction depicted in Figure 2(a), by combining knowledge about the request (the *L1 REQ C* tells us this is an upgrade for address *a*) and the current state of *a* (cached in L1'), the exact coherence transaction can be determined. Once the coherence transaction is known, the next messages can be predicted based upon the previous message. To implement this predictor structure, a 5-state Finite State Machine (FSM) is used per address with the state transitions based upon all possible coherence transactions. The FSM is designed for the MESI protocol but could be modified for other coherence protocols. At runtime, every message that travels the network is intercepted and used to make the state transitions. Future messages are predicted by looking at the new state and the last observed message. There are 5 possible states:

State 1 block is not present in the on chip cache hierarchy and has been requested by the directory/L2

State 2 block is only present in the L2

State 3 block is only present in one L1 cache

State 4 block is present in multiple L1 caches

State 5 transient state, block is in the midst of being evicted

Figure 2(c) shows the state machine transitions for the ST access depicted in Figure 2(b). In the conventional case, all 5 messages in the transaction are delayed by the path setup latency before transmission and path setup only starts after the controllers have handled the previous message. Using the predictor, as soon as a message is intercepted, a prediction can be made to setup the path for the next message while the message is still being handled by the cache/directory controller. Only the first message in the transaction is delayed by the path setup latency.

The predictor setup consists of a look-up table (LUT) in which every entry is associated with one unique address. Every entry has 5 fields: (1) a valid bit, (2) the current FSM state (the first and second field can be combined into 3 bits), (3) field tracking L1 caches holding a copy of the address (N bits for an N-core CMP), (4) the type of transaction (3 bits for 5 possibilities: LD, ST, Upgrade, L1 Eviction and L2 Eviction) and (5) the L1 cache currently upgrading or evicting an address ($log_2(N)$ bits). The LUT is located next to the central allocator. The address of the intercepted message is used to address the LUT and (if needed) issue a speculative path request to the allocator. Real path requests will be prioritized over speculative request by the allocation algorithm. If the speculative request wins a path, the corresponding source node will be notified of the path that has been setup. If the predictor did not predict the correct path, the message will go through the normal request-grant cycle, without any additional penalties. Previous work has also shown that the shared memory networks are typically operated at very low load ($<$1%) so unused speculative requests are unlikely to overload the allocator [8].

IV. Architecture

This work is aimed at a distributed, shared memory CMP where the various caches are kept coherent using a directory protocol. Directory protocols are the most scalable solution for higher core counts [14] but exacerbate the optical setup overhead as all L1 misses go via the directory in the form of short coherence messages. In this work we assume a CMP with 16 tiles with each tile consisting of a CPU, private L1 (64 kB, 2-way associative) and part of the logically shared, physically distributed L2 (16 $\times$128 kB, 8-way associative) and directory. All the tiles are connected via an optical crossbar as shown in Figure 3 with wavelength striped links to reduce serialisation latency. Logically an optical crossbar (as with its electronic counterpart) provides all-to-all connectivity provided the switch has been setup to connect the requested inputs and outputs and could be implemented using various physical technologies including ring resonators and Mach-Zehnder interferometers. Each tile has control links directly to the allocator, implemented as $log_2(N) + 1$ parallel electrical wires ($log_2(N)$ for destination field and 1 valid bit) for fast signalling of requests and grants, where N is the number of tiles. The predictor needs to take all messages traversing the

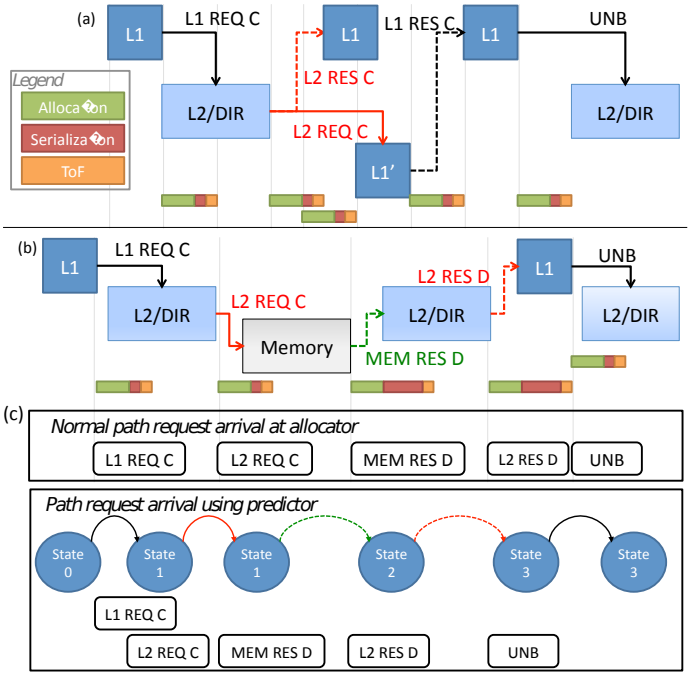

Fig. 2. Examples of memory transactions and the messages generated. The timescales in this figure only show the relative order of events and not the duration of these events. (a) an L1 upgrade transaction (b) a write access to an address not present in the on-chip cache hierarchy. Below the messages, their corresponding stages in the network are depicted: path allocation, serialization and time of flight (ToF). (c) Resulting transitions in the predictor FSM

TABLE I
COMPARISON OF LATENCY WITHOUT CONTENTION

Network Type	Control Latency (clock cycles)	Data Latency (clock cycles)
Crossbar with correct prediction	4	17
Crossbar without prediction	12	25
SWMR 1 wavelength	15	117
SWMR 3 wavelengths	7	40
Mesh (16 cores)	12	28

network into account. Hence, as shown in Figure 3, there is a coupler at the input to each crossbar port which taps off a small proportion of the optical signal to intercept the message. This doubles the number of receivers but the additional optical power required is very small as the predictor receivers are positioned before the crossbar switch which dominates the loss budget for the link.

For each port, 8 wavelengths are assumed with each wavelength modulated at 10Gb/s giving 2 clock cycles serialisation latency. Assuming a clock frequency of 2 GHz and a die size of 400 mm^2, the time of flight latency for signals in the photonic network is also 2 clock cycles. The path setup latency (if required) is 8 clock cycles consisting of 2 cycles each to send requests and grants and 4 clock cycles for arbitration [8]. The latencies of the NoCs used for comparison can be found in Table I

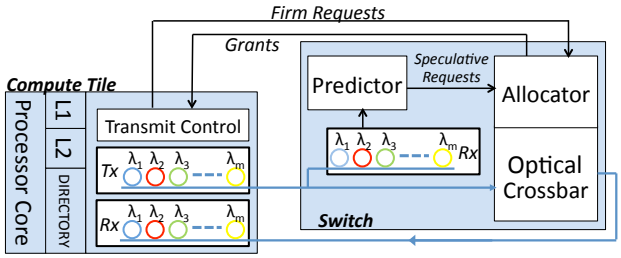

Fig. 3. Proposed network architecture which provides low serialisation latency through wavelength striping and allows the central predictor to observe coherence traffic. For clarity, the connections between only one compute tile and the central switch are shown. All other tiles connect to the switch in the same way.

V. RESULTS

The predictor implementation has been tested using traces obtained with the full-system, cycle accurate computer system simulator gem5 [15], running various benchmarks from the PARSEC benchmark suite [16]. The traces are taken in the parallel phase of computation (region of interest) and capture all messages traveling the NoC until 1 million instructions have been committed. The NoC has been implemented as a point-to-point network where every cache/directory controller has a dedicated network node so there is no interference between messages, caused by the network. Figure 4 shows the prediction hit rates for the *x264* benchmark. This shows the predictor achieves very high hit rates: overall, over 85% of all messages are predicted. However, the hit rates do differ for the various message types. The hit rate for *L1 REQ C* is around 50% even though the predictor does not aim to predict these messages as they signal a L1 miss. This can be explained by the principle of spatial and temporal locality as exploited in [8]. In principle, perfect prediction of all messages excluding the ones that signal the start of a transaction is possible if the predictor is accurately designed whereupon it basically becomes a directory controller without the actual caching. However, in our scheme the predictor can only track one concurrent transaction so multiple transactions for the same address starting around the same time will not be predicted correctly. Tracking concurrent transactions would reduce miss rates but would increase the complexity of the predictor.

Figure 5 shows the prediction hit rates for other benchmarks in the PARSEC benchmark suite. With a perfect predictor the hit rates should be the same as the prediction is based on the coherence protocol which remains the same. However, the number of concurrent transactions and the types of messages transmitted differs per benchmark resulting in variations in hit rates in practice: *blackscholes* for example has the lowest prediction hit rate but has the highest number of *L1 REQ C* messages which are not predicted. The effect of prediction on the overall performance will also differ for the various benchmarks as they have different computation-communication relationships. To get a first indication of the effect of prediction, average latencies were calculated per message type per benchmark using the prediction hit rate results

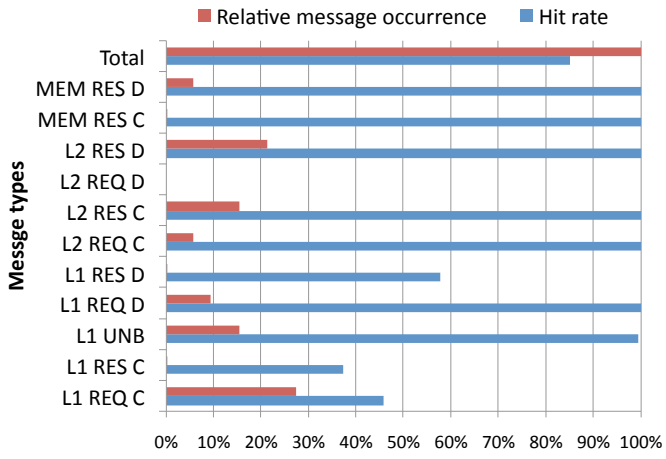

Fig. 4. Prediction hit rates and relative number of occurrences for the various message types in *x264*. *L2 REQ D* messages are writebacks from the L2 to main memory and do not occur during this period.

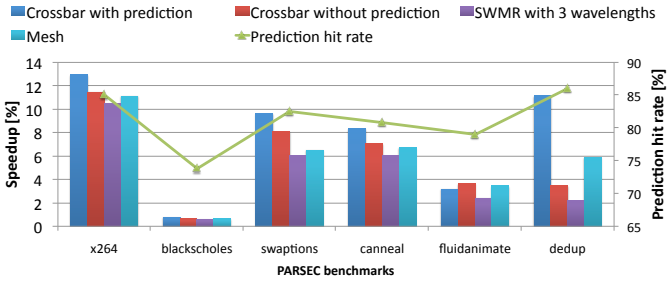

Fig. 5. Speedups of PARSEC benchmarks for various topologies, normalised to a SWMR scheme with 1 wavelength. The line curve shows the prediction hit rate

and used in full system gem5 simulations and were compared with the average message latencies for two SWMR schemes with 1 and 3 wavelengths respectively, an optical crossbar without prediction and an electrical mesh as shown in Table I. Figure 5 shows the performance for various benchmarks on different topologies, normalised to the SWMR scheme with 1 wavelength. The optical crossbar schemes outperform the SWMR scheme in all cases due to the higher bandwidth per port and hence lower serialisation latency. Adding the predictor to the crossbar increases the speedup by up to 7.7 % for *dedup*. The increased benefit in the case of *dedup* is due to the high occurrence of shared writes in this benchmark leading to a higher number of memory transactions involving invalidations. These longer transactions with more messages benefit more from the prediction scheme. The predictor offers speed ups of 1.5%, 1.6% and 1.3 % for *x264*, *swaptions* and *canneal* but no significant benefit for low communication benchmarks such as *blackscholes* and *fluidanimate*. The 16-core mesh network, for which we chose fairly aggressive parameters, performed well in all cases. However, mesh latency is very sensitive to hop count [17] and the relative performance would not be sustained with higher number of cores.

VI. Conclusion

The results in this paper show that the path setup overhead in switched optical NoC can be effectively hidden by using coherence based prediction. We have shown that up to 85 % of messages can be predicted, leading to up to 7.7 % speedup in application performance in the benchmarks tested, thereby outperforming non-switched optical NoCs. The size of the predictor is determined by the number of addresses kept in the LUT. The directly mapped LUT investigated in this work will have an unreasonably large area but only a limited number of addresses in the LUT will be used. Preliminary results have shown that by using a set-associative strategy as used in caches, the size of the LUT can be reduced to the size of one slice of the distributed L2 and the predictor can still obtain a hit rate over 75 % in the case of *x264*. However, further work is needed to find optimal mapping strategies and investigate the complexity, power consumption and latency of the predictor and its integration into the NoC.

Acknowledgements

This work was supported by the UK Engineering and Physical Sciences Research Council (EPSRC) Fellowship grants to Philip Watts (EP/I004157/2) and Timothy Jones (EP/K026399/1).

References

[1] D. Wentzlaff *et al.*, "On-chip interconnection architecture of the Tile processor," *Micro, IEEE*, vol. 27, no. 5, 2007.
[2] J. Shin *et al.*, "A 40 nm 16-Core 128-Thread SPARC SoC processor," *JSSC*, vol. 46, no. 1, 2011.
[3] A. Shacham and K. Bergman, "Building ultralow-latency interconnection networks using photonic integration," *Micro, IEEE*, vol. 27, no. 4, july-aug. 2007.
[4] D. Vantrease *et al.*, "Corona: System implications of emerging nanophotonic technology," in *Proc. ISCA*, June 2008.
[5] A. Krishnamoorthy *et al.*, "Computer systems based on silicon photonic interconnects," *Proc. of the IEEE*, vol. 97, no. 7, july 2009.
[6] Y. Pan *et al.*, "Firefly: Illuminating future network-on-chip with nanophotonics," in *Proc. ISCA*, 2009.
[7] G. Hendry *et al.*, "Time-division-multiplexed arbitration in silicon nanophotonic networks-on-chip for high-performance chip multiprocessors," *J. Par. and Dist. Comp.*, vol. 71, no. 5, 2011.
[8] M. R. Madarbux *et al.*, "Towards zero latency photonic switching in shared memory networks," *Concurrency and Computation: Practice and Experience*, 2014.
[9] P. Watts *et al.*, "Energy implications of photonic networks with speculative transmission," *J. Opt. Comm. and Netw.*, vol. 4, no. 6, 2012.
[10] M. Acacio *et al.*, "Owner prediction for accelerating cache-to-cache transfer misses in a cc-NUMA architecture," in *Proc. Supercomputing*, 2002.
[11] S. Kaxiras and C. Young, "Coherence communication prediction in shared-memory multiprocessors," in *Proc. HPCA*, 2000.
[12] M. Martin *et al.*, "Using destination-set prediction to improve the latency/bandwidth tradeoff in shared-memory multiprocessors," in *Proc. ISCA*, 2003, pp. 206–217.
[13] C. Adi *et al.*, "An efficient path setup for a photonic network-on-chip," in *Proc. ICNC*, nov. 2010.
[14] M. M. K. Martin *et al.*, "Why on-chip cache coherence is here to stay," *Communications of the ACM*, vol. 55, pp. 78–89, July 2012.
[15] N. Binkert *et al.*, "The gem5 simulator," *SIGARCH Comput. Archit. News*, vol. 39, 2011.
[16] C. Bienia *et al.*, "The PARSEC benchmark suite: Characterization and architectural implications," Princeton University, Tech. Rep. TR-811-08, January 2008.
[17] W. Dally and B. Towles, *Principles and Practices of Interconnection Networks*. Morgan Kaufmann Publishers Inc., 2003.