Towards Zero Latency Photonic Switching in Shared Memory Networks

Anouk Van Laer,
Muhammad Ridwan Madarbux,
Philip M. Watts
Dept. of Electronic and Electrical Engineering
University College London
{anouk.laer.11,m.madarbux,philip.watts}
@ucl.ac.uk

Timothy M. Jones
Computer Laboratory
University of Cambridge
timothy.jones@cl.cam.ac.uk

ABSTRACT

Photonic networks-on-chip based on silicon photonics have been proposed to reduce latency and power consumption in future chip multi-core processors (CMP). However, high performance CMPs use a shared memory model which generates large numbers of short messages, creating high arbitration latency overhead for photonic switching networks. In this paper we explore techniques which intelligently use information from the memory hierarchy to predict communication in order to setup photonic circuits with reduced or eliminated arbitration latency. Firstly, we present a switch scheduling algorithm which arbitrates on a per memory transaction basis and holds open photonic circuits to exploit temporal locality. We show that this can reduce the average arbitration latency overhead by 60% and eliminate arbitration latency altogether for a significant proportion of memory transactions. We then show how this technique can be applied to multiple-socket shared memory systems with low latency and energy consumption penalties. Finally, we present ideas and initial results to demonstrate that cache miss prediction could be used to set up photonic circuits for more complex memory transactions and main memory accesses.

Categories and Subject Descriptors

B.3.2 [**Memory Structures**]: Design styles—*cache memories, shared memory*; B.4.3 [**Input/Output and Data communications**]: Interconnections; C.1.2 [**Processor architectures**]: Multiple Data Stream Architectures (Multiprocessors) —*Interconnection architectures*

General Terms

Design, Measurement, Performance

Keywords

Photonic interconnection networks; Networks-on-chip; Shared memory architectures

Exploiting Silicon Photonics for Energy-Efficient Heterogeneous Parallel Architectures (SiPhotonics'2014) Vienna, Austria

1. INTRODUCTION

Photonic networks on chip (NoC) based on advances in silicon photonics have been widely proposed as one of the solutions to the serious problems of energy consumption and thermal management in chip multiprocessors (CMP) [1–6] due to the fundamentally lower power consumption of photonic communication [7]. In addition, photonic communication enables high bandwidth end-to-end network paths for global on-chip paths or for systems spanning multiple chips without significant power penalties. Figure 1(a) shows a current typical 4-socket high performance shared memory server architecture based on [8]. Due to the fundamental difference between electronic communications for on-chip (wide buses of small wires) and off-chip (serial transceivers driving transmission lines), separate networks are used for on-chip and chip-to-chip communications with the architecture constrained by the limitations of the electronic interconnect. Furthermore, the SERDES used in off-chip communications consume >20% of total chip power [8]. By contrast, there is no fundamental difference between photonic on-chip and off-chip links, allowing us to build single unified low latency photonic networks, as shown in Figure 1(b), to increase performance of shared memory systems spanning multiple chips, or even boards and racks.

NoCs in current systems consist of electronic crossbars [8] or meshes [9] relying on multiple hops between sequential elements. However, photonic NoC require end-to-end optical paths to be set up in advance of communication and the resulting latency overhead of arbitration and control message transmission between cores and a central switch can be significant. Figure 2 shows the sources of latency in a scheduled photonic switch. Setting up an optical path involves sending a request to the switch arbiter, performing arbitration and returning a grant to the requesting port. We label this time between the transmission of the optical path request and the actual start of the optical transmission, the arbitration latency. The head latency is the time taken for the head of the message to be received at the destination port and includes serialization and deserialization times as well as the time of flight in the waveguide. Note that head latency also applies to the request and grant control messages. Synchronisation latency can be neglected in NoC in which the transmitter and receiver share the same clock but can be significant in chip-to-chip networks - we discuss this issue further in the

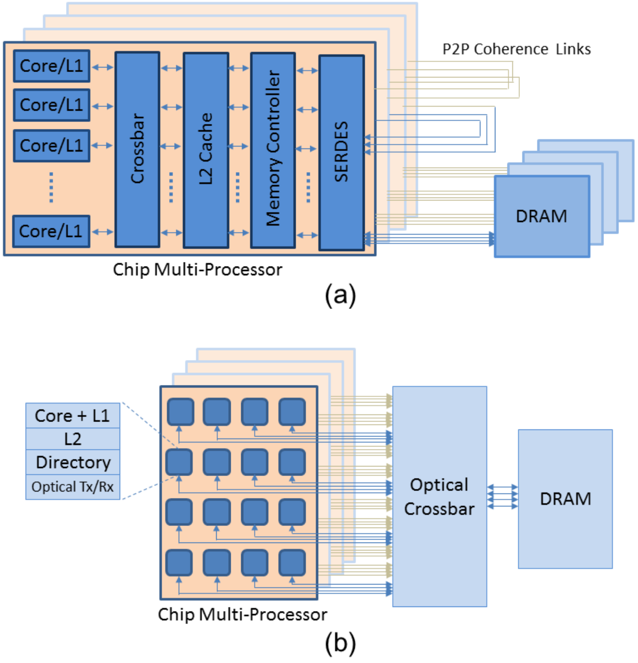

Figure 1: High performance multi-socket servers (a) current architecture with NOC and point-to-point chip-to-chip SERDES links (b) future architecture with unified switched photonic network.

conclusions. Data serialization latency can be very low if a broadband switch is used and messages are wavelength striped to use the high bandwidth of photonic links (high bit rate and multiple wavelengths per waveguide).

This paper focuses on the question of reducing or eliminating arbitration latency. As the majority of traffic in a shared memory system consists of short (8–256 B) coherence messages between caches and directory controllers, arbitration latency can impose a high overhead. Various proposed schemes for overcoming this latency overhead are reviewed in Section 2, but all involve an increase in the number of optical components and/or the complexity of the control plane. In contrast we explore techniques for eliminating arbitration latency by prediction of communication within shared memory systems. Prediction already plays a major role in increasing the performance of modern computer architectures, for example through branch speculation or prefetching in cache hierarchies. The prediction techniques discussed here could also be used in electronic networks, but, due to the hop by hop communication nature of meshes or highly pipelined crossbars, they will have lower impact than in future silicon photonic networks.

The rest of the paper is organised as follows: following the review of previous work in section 2, we describe the shared memory system assumed in this work and its communication characteristics in section 3. Section 4 reviews our previous work (first presented in [10]) on the latency benefits of arbitrating for memory transactions rather than individual messages and presents new results showing the performance improvements for multiple socket systems like the one shown in Figure 1(b). Section 5 takes this concept a step further by exploring the concept of setting up photonic paths in advance by prediction of cache misses. Initial results are presented. Finally section 6 discusses the system implications of these results and further work.

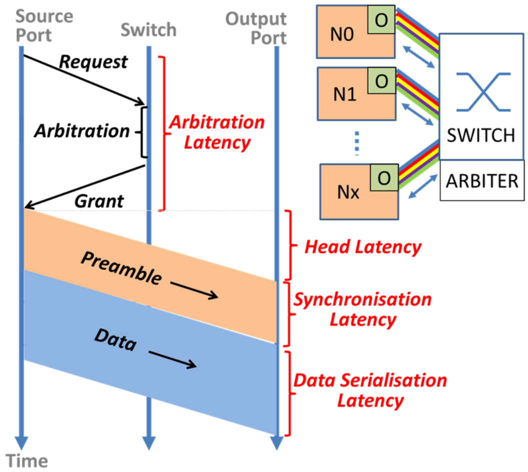

Figure 2: Sources of latency in a wavelength division multiplexed photonic switch.

2. PREVIOUS WORK

In this section we review techniques proposed for reducing control and arbitration latency in photonic computer networks. Speculative transmission, in which messages are transmitted before a grant has been received and either dropped or redirected if there is no path available, has been proposed, either operating in parallel with a centralized arbiter (OSMOSIS [11]) or independently (SPINet [1]). Speculative transmission forces the use of strict time slots and, used independently, suffers from reduced maximum throughput and head of line blocking. High performance speculative schemes also require the additional complexity of reordering in the receiver [11, 12]. SPINet [1] also reduced arbitration latency using a distributed arbitration scheme consisting of a separate wavelength transmitted with the data to determine the configuration of each switching stage, whereas, CORONA used an optical token ring arbitration scheme [2]. The single writer multiple reader (SWMR) topology adopted by Firefly [4] avoids arbitration altogether by allowing each node to receive from all other nodes simultaneously, but requires flow control to avoid receiver buffer overflow and a reservation scheme for acceptable power consumption. Oracle's Macrochip [3] also avoids arbitration using a wavelength and space division multiplexed all-to-all network, but suffers from high serialisation latency compared with wavelength striped approaches. Both SWMR and all-to-all topologies require greatly increased number of optical components compared with a basic crossbar. Other architectures reduce the arbitration overhead by splitting up the network into smaller photonic switch sections interspersed with optical-electrical-optical (OEO) conversions to allow electronic buffering, for example [6] in which routing in the x and y directions of an optical mesh are handled separately. However, these schemes reduce the power consumption and latency benefits of introducing photonic networks.

In contrast to the packet switched networks discussed above, the use of relatively long-lived optical circuits to provide low latency transmission of long lived flows or large messages has been investigated in the context of supercomputers [13] and a torus NoC [5]. In this case it is usually necessary to have a backup electronic network to carry small messages. For shared memory systems, the authors have investigated the concept of setting up long lived circuits ($\gg$ message length) between cores which have dense memory sharing requirements. Initial results [14] showed that, with ideal circuit setup decisions made on less than 1 μs time periods, a large proportion of traffic from PARSEC applications could be routed onto the circuit switch. However, further investigation has shown that adding background traffic from the operating system considerably reduces the benefits. In addition, overall power consumption is dominated by the backup electronic network, so the power savings from adding the optical circuit switch are proportionally small.

In contrast to the above, this paper discusses techniques for intelligently setting up optical paths by predicting network communication using information from the memory hierarchy. For NoCs, various prediction schemes have been proposed to reduce the latency of the average memory request. In [15], the need for cache-to-cache transfers are predicted based upon the program counter, while caches holding copies of the requested data are predicted using both the program counter and the requested memory address. In [16] prediction is used to forward memory addresses to future readers, thus avoiding L1 misses and the following indirection to the directory. In [17] a cache coherence protocol is proposed which forms a hybrid between a directory and snooping protocol. Coherence messages are forwarded to the predicted sharers of a block (destination-set) and the home node. The home node holds a directory structure which compares the predicted destination set with the actual sharers. While these proposals decrease the latency of memory request by avoiding unnecessary network transactions, they do not speedup the messages that still need to travel the NoC.

Other prediction schemes make decisions based upon events in the network. In [18] 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 [19], flow control is acheived by predicting congestion in the network and hence controling the injection rate.

3. METHOD

The system we assume for all the results presented in this paper (see Figure 1(b)) consists of 32 tiles connected using an optical crossbar. Each tile contains an in-order x86 processing core, a private L1 cache (16 kB for instructions,16 kB for data), part of the shared L2 cache (1 MB in total) and part of the directory. The MESI cache coherence protocol is used to keep the physically distributed memory coherent. Coherence messages of 8B for control messages and 72B for data messages (8B + 64B cacheline) are exchanged between the tiles over a central optical switch, for example using microring resonators [20], with 1 optical port per tile. For a single network-on-chip with a clock frequency of 2 GHz and a die size of 400 mm^2, the worst case optical time of flight between any two ports communicating over silicon waveguides with $n_{eff} = 4.2$ is less than one clock cycle. Including serialisation and other circuit delays, we can conservatively assume a maximum of 2 clock cycles for the head latency and request/grant tranmission. However, for mutliple chip systems on a single PCB such as that shown in Figure 1b with a maximum distance between port and switch of 0.5m communicating over polymer waveguides or optical fibre with $n_{eff} = 1.5$ the head latency would be up to 7 clock cycles.

Trace files, containing all the coherence messages travelling the network, were generated using the cycle accurate, full system simulator gem5 [21] which is able to boot Linux and run the PARSEC benchmark suite [22]. This benchmark suite contains a collection of financial, animation, routing algorithm, compression, server search and online clustering algorithms which provide a realistic workload for a CMP. To remove the effect of the network from the traces, ideal contention free interconnects were implemented in the simulation.

The measure of performance in this work is the Average Memory Access Time (AMAT) which is a good indicator of systems performance for in-order cores [23]:

$$AMAT = \mathit{Hit\ time} + (\mathit{Miss\ Rate} \times \mathit{Miss\ Penalty})$$

The hit time is defined as the time taken to satisfy a memory request by a core if the requested memory address is available in the L1. In the case the block is either not present or the L1 cache does not have the correct permissions, a miss occurs. The miss penalty is defined as the time taken to correct this situation by either fetching the block or obtaining the permissions needed.

The work presented in this paper exploits the fact that messages in a shared memory network are generated in sequences initiated by transistions in the cache coherence protocol in response to memory requests from the cores. Figure 3 shows some examples of coherence message sequences which commonly occur in the MESI protocol showing examples of memory transactions which: (b, c) involve just two tiles and hence can be served by a bidirectional optical path; (a) involve three or more tiles which require additional optical paths and (d) involve communications with main memory. We use knowledge of these transactions to efficiently set up optical paths (or circuits) between tiles and main memory.

Figure 4 shows the variation in occurrence and average latency of coherence message sequences of different lengths. The length of a message sequence is defined as the number of messages (transmitted on the NoC) needed to complete a coherence transaction. The average latency (Figure 4(a)) depends both on the sequence length and whether or not main memory is involved. The occurrence of each sequence (Figure 4(b)) differs depending on the communications requirements of individual benchmarks. Figure 4(c) shows the resulting weighed latency. The latencies of sequences consisting of 5 or more messages might be longer than pictured as these sequences are often coherence transactions involving the invalidation of memory addresses shared by multiple caches making the latency determined by the number of sharers. Figure 4 shows the lower bound where there is only one other sharer in the system. This figure shows the AMAT can be reduced by either focussing on the most common sequences (with a length of 2 or 3 and no main memory access) or the sequences with longest latencies (sequences involving main memory accesses or consisting of 5 or more messages).

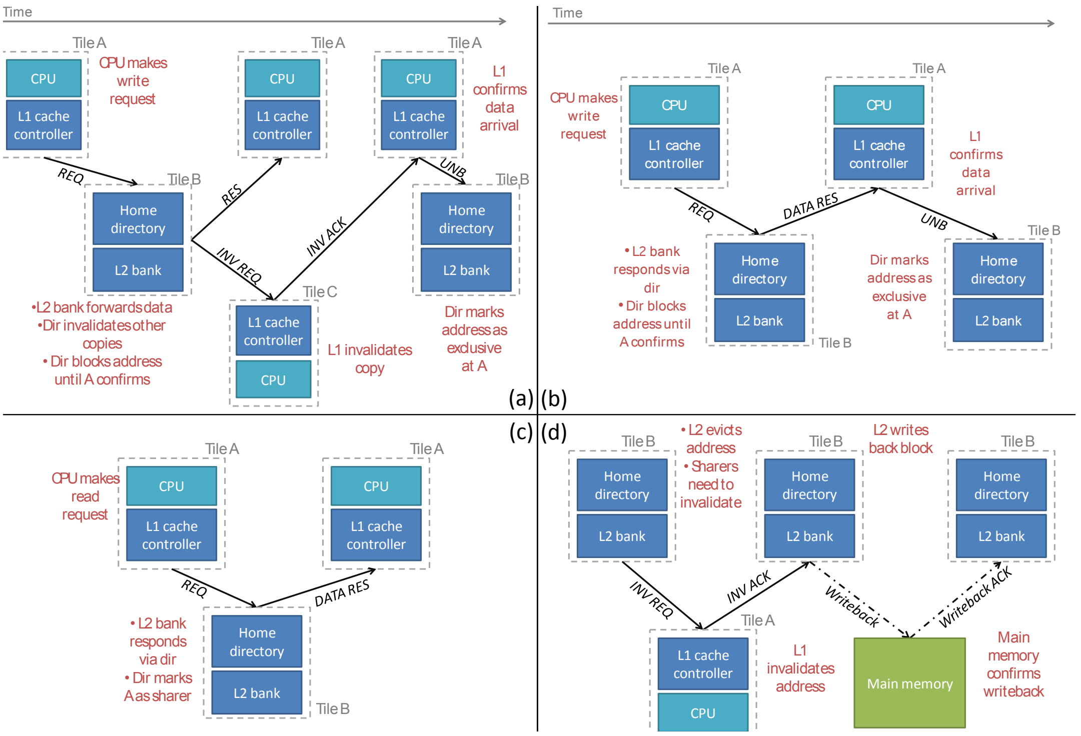

Figure 3: Examples of common coherence message sequences in the MESI protocol (a) CPU A requests *store* access to a memory address cached in other L1 caches (b) CPU A requests *store* access to a memory address cached only in the L2 (c) CPU A requests *load* access to a memory address (d) L2 evicts a block which is cached in a private L1

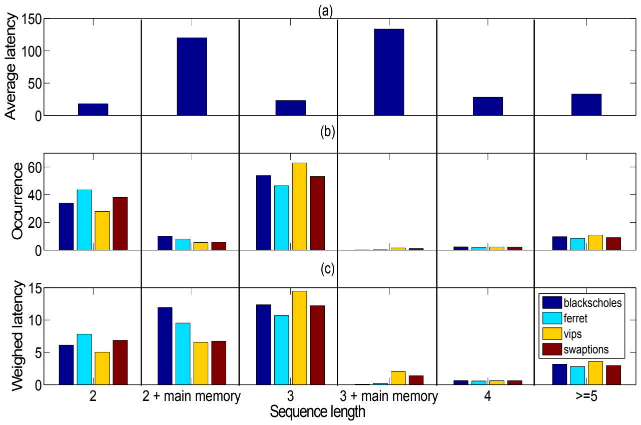

Figure 4: (a) Average latency, (b) probability of occurrence and (c) weighed latency (average latency×occurrence) for memory transactions. Latencies are in clock cycles.

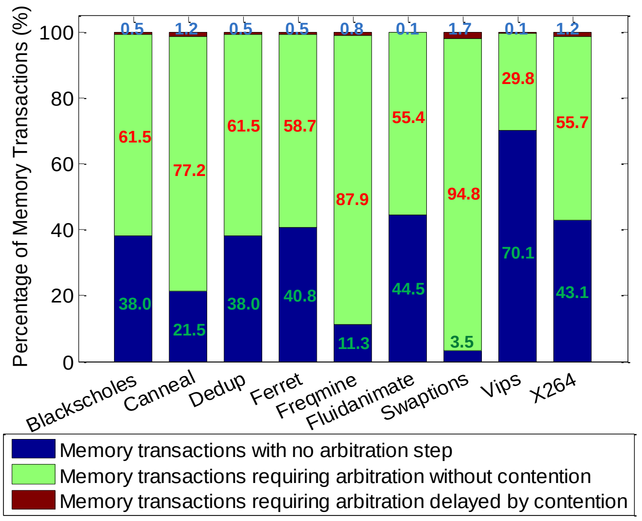

Figure 5: Arbitration outcomes for PARSEC benchmarks using the arbitration per memory transaction algorithm

4. ARBITRATION PER MEMORY TRANSACTION

In the discussion of arbitration latency in section 1 we assumed that each message goes through the request, arbitration and grant process. This would be appropriate for random and independent messages without temporal or spatial locality. However, in a shared memory coherence network messages are communicated based on the cache coherence protocol finite state machine as shown in Figure 3. For transactions involving just two cores such as the examples in Figure 3 (b), (c) and (d), all the information required to set up the bidirectional optical paths is available in the Miss Status Holding Register (MSHR) at the source port. Thus, arbitration only needs be performed once for the initial request message, leaving the optical circuits open for subsequent messages in the same memory transaction, reducing latency compared with arbitrating for every message. In our previous work using this technique [10], we showed that the arbitration latency overhead for PARSEC benchmarks running on a single chip 32-core x86 system can be reduced by between 31.8 % and 70.6 % depending on the distribution of message sequence lengths and the amount of sharing between cores in the benchmark.

There are however two drawbacks of keeping circuits open for extended periods of time: (1) additional energy is consumed in the switches and (2) other communications targeted at either of tiles involved in a memory transaction must wait until the transaction is complete (whereas if arbitration is taking place per message, the communications can be interleaved). On the energy question, for a network with off-chip WDM "photonic power supplies", we find that the power required to keep the optical switches in the on state [20] is negligible as compared to the power dissipated on the chip by the external laser source and receiver. Maintaining the optical circuits for an extended period of time would therefore only marginally affect the overall energy consumption. On the question of latency, we showed in [10] that only a very small proportion of individual messages have increased latency due to circuit contention. This is because the PARSEC benchmarks, as with other applications, load the network very lightly [24]. Given these two points, it makes sense to hold open circuits for the current memory transaction to complete. In addition, using the principles of temporal and spacial locality, it is likely that subsequent memory transactions will involve the same two cores, so optical circuits can be held open unless another request is made to either of the cores. Figure 5 shows the variation of the percentages of memory transactions which benefit from the circuit remaining open for the different benchmarks considered together with the percentage experiencing contention. In the case of `vips`, 70.1 % of memory transactions benefit from no arbitration overhead latency. The proportion of messages experiencing contention is <2 % for all benchmarks. However, for the multiple chip systems shown in Figure 1b, circuits must be held open for longer to accommodate longer time of flight latencies (see Section 3) thus increasing the blocking probability as well as the head latency of request, grant and data messages. Below, we present new results quantifying the performance benefits for larger shared memory systems when keeping circuits open for a whole transaction, compared with arbitrating for each message.

Figure 6 shows that per transaction arbitration has greater latency benefits in absolute terms for networks with longer time of flight between tile and switch, although the percentage decrease in the average arbitration overhead per message remains quite constant at 60 % and there is greater variation between benchmarks. In addition, it can be observed from the nearly linear relationship between the arbitration overhead and time of flight that there is no significant increase in contention due to the increased memory transaction times.

As can be seen from Figure 7, there is a decrease in AMAT for all values of time of flight when using arbitration per transaction as compared to arbitration per message. The effect of the arbitration algorithm is more pronounced in this analysis by showing that for a time of flight of one clock cycle, the decrease in the AMAT amounts to only 3.2 % as compared to a decrease of 14.3 % when the time of flight is ten clock cycles. The linear relationship suggests again that contention does not affect the results significantly. As would be expected, benchmarks such as X264 with high communication requirements benefit more from per transaction arbitration.

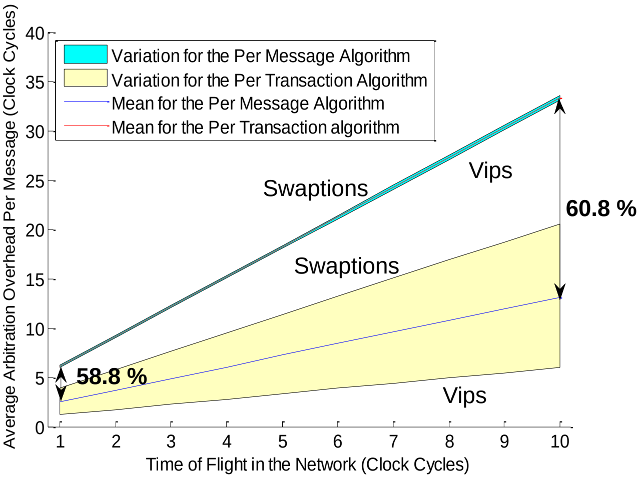

Figure 6: Variation of the average arbitration overhead per message with time of flight between tile and switch

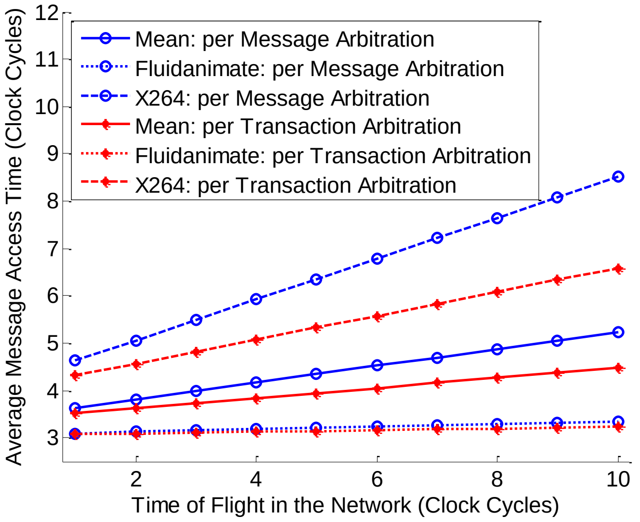

Figure 7: Variation of the average memory access times with time of flight between tile and switch

5. PHOTONIC NETWORK SETUP THROUGH CACHE MISS PREDICTION

The technique described in the previous section reduces arbitration latency by setting up bidirectional circuits based on a knowledge of the communications produced by the MESI protocol. However, arbitration for new circuits cannot begin until the request has reached the MSHR and more complex transactions (such as those shown in Figure 3(a) and (d)) will require two or more arbitrations. Across all the PARSEC traces studied, 16% of all transactions take

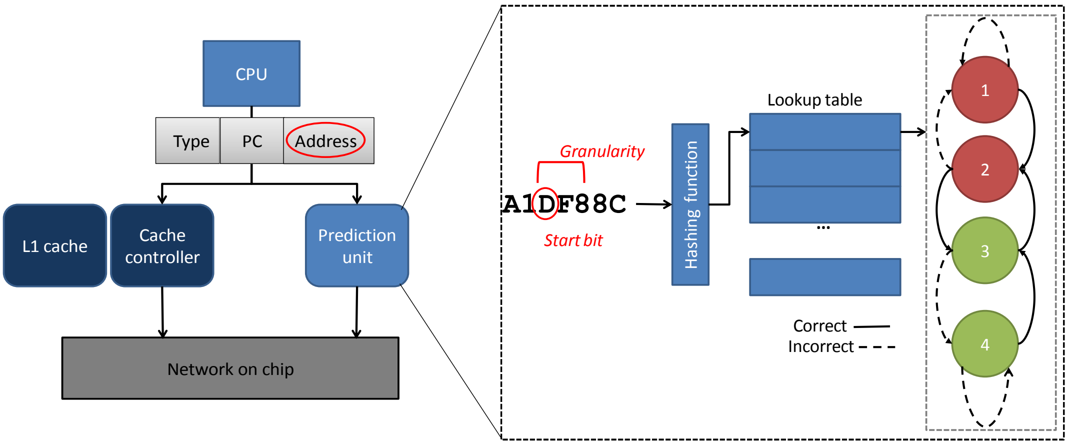

Figure 8: Operation of the prediction unit and interface with the tile.

place between three or more tiles and transactions involving main memory access, although relatively rare, have a high impact on AMAT. In this section, we explore the possibility of further latency savings through cache miss prediction. The work in this section is preliminary. The prediction algorithm has only been tested on the `blackscholes` benchmark.

A first step towards the prediction of exact message sequences based upon information from the cache controllers is the prediction of coherence requests leaving the L1 cache using a local predictor operated in parallel with the cache access. If a coherence message is predicted, a path request will be send out to the central arbiter to setup the required optical paths before the actual coherence message reaches the network interface. While this scheme is very easy to implement, the predictor should be faster than the cache access in order to reduce latency. As L1 caches are geared towards low latency operation (1–3 clock cycles), we believe using a predictor to solely setup optical paths for cache coherence messages leaving L1 caches is suboptimal.

Because of the latency constraints imposed on the L1 predictor, we wish to be able to predict messages that cannot be serviced directly by the L2 bank associated with the directory, for example a cache-to-cache transfer (Figure 3(a)) or a main memory access (Figure 3(d)), based upon the information in the memory request leaving the CPU. As a first step, we only predict the existence of such a coherence message but not its destination or message type. While the idea behind this predictor is the same as in the L1 predictor case, the actual implementation is more complex as the feedback needed to update the predictor will need to come from a different node. One possibility is piggybacking the outcome of the prediction on coherence messages traveling to the home node of the predictor.

The prediction is made based upon the memory address requested (address based prediction), operating in parallel with the L1 cache access. The predictor used in this work (Figure 8) consists of a lookup table (LUT) with N entries and some peripheral circuits to convert (part of) the address into a key for the LUT, update the entries in the LUT and send out circuit setup requests based upon the prediction that was just made. The lookup table is accessed by hashing a proportion of the requested memory address. Every entry in the lookup table consists of the state of a 2-bit counter, the last prediction, a valid bit and in the case of a set-associative organization a tag. In state 1 and 2 no message will be predicted whereas in state 3 and 4 a message will be predicted. After the prediction the state of the 2-bit counter will be updated based upon the correctness of the prediction. The memory address consists of 7 hexadecimal digits. Using the complete memory address to obtain a key is inefficient as 256M keys would be possible. Sweeping over the *granularity* (number of hexadecimal digits used to obtain the address) shows a higher granularity will result in a lower percentage of messages for which no optical path was setup which comes at the cost of a larger LUT as can be seen in Figure 9.

To find the digits in the memory address that carry most of the information, we investigated the effect of the *start digit*. This is the first digit to be included in the address hashing. As Figure 9 shows the various bits in the address do not contain the same information. By carefully choosing the correct start digit and keeping the granularity the same, the missed message rate can be reduced by more than 70%. The three least significant digits of an address (marked in Figure 9 as start digit 5,6 and 7) do not carry a a lot of information. This can be explained by the fact the page size is set to 4KB and so these three digits form the page offset.

Although the size of the LUT is quite large, most of the entries are never used: for a granularity of 3 digits and higher less than 30% of the entries are used. This decreases to less than 0.001 % for a granularity of 6. We can reduce the size by changing from the directly mapped setup to a set-associative organization which has a beneficial effect on the misprediction rate as shown in Figure 10. The LUT size of the set-associative predictors was set to 256 entries. For comparison, this is the size of a directly mapped predictor with a granularity of 2. When evicting one address from the LUT, this entry will be reset. The start state of the 2-bit saturating counter is state 4 in which a message will be predicted. By evicting entries from the LUT, the LUT gets slightly biased towards predicting more messages. Increasing the set-associativity will increase the latency of the predictor though as more entries need to be searched. A careful trade-off between the latency and size of the predictor will need to be made.

These first results are encouraging: in a directly mapped

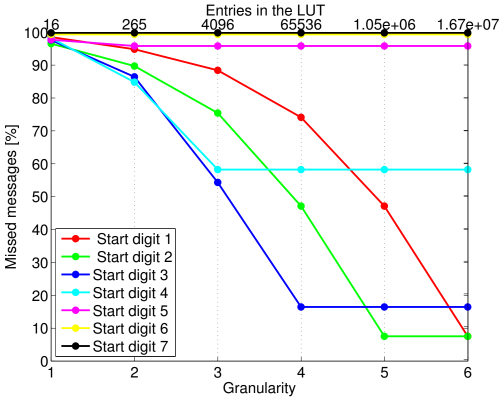

Figure 9: Percentage of coherence messages for which no optical path was predicted.

setup, using a granularity of 4 (resulting in a LUT of 64K entries) and the start digit set to 3, only 16% of the coherence messages leaving the an L2 bank will not have an optical path setup in advance. By using a 8-way set-associative setup with start digit 4 and granularity 4 but a table size of only 256 entries, the number of unpredicted messages can be reduced to less than 25%. There is a drawback to this simplistic predictor though. The number of optical paths that will be setup but never used is inversely related to the number of optical paths that will be used by a coherence message. In some combinations of granularity and associativity, 70% of setup paths are not used. As the network load is low, as discussed in Section 4, setting up unused optical paths is not necessarily a problem but we still wish to reduce this number by improving the existing predictor.

Further work is required to setup optical paths based upon these predictions which can be subdivided into two parts. Firstly the predictor needs to be optimized. The predictor discussed in this work is able to predict off-chip main memory accesses or the start of a cache-to-cache transfer. However, to setup a correct optical path for a cache-to-cache transfer, the nodes accessed by this transaction need to be known. To achieve this, a global predictor will be combined with a local predictor. The local predictor will predict the cache-to-cache transfer while the global predictor will use information present in the central arbiter to predict the nodes that will be addressed in this transaction. The predictor used is address based but a hybrid predictor that combines the memory address and the program counter causing the memory access might give better performance. The distribution of the keys in the LUT needs to be optimized as the LUT is underutilized. The second step will be the implementation of these prediction schemes into gem5 [21]. We have already extended this simulator with optical network models [25]. By also implementing the predicting schemes into gem5, we can give more definite answers towards its effect on the performance of the overall CMP.

6. CONCLUSIONS

In this paper, we have presented techniques that can significantly reduce the arbitration latency of photonic networks for future shared memory computer systems. Firstly, we have demonstrated that a switch scheduling algorithm which arbitrates on a per memory transaction basis and holds open photonic circuits to exploit temporal locality can reduce the average arbitration latency overhead by 60 % and eliminate arbitration latency altogether for a significant proportion (> 70 % for vips) of memory transactions. We have also presented ideas and initial results to show that cache miss prediction could be used to setup photonic circuits for more complex memory transactions and main memory accesses.

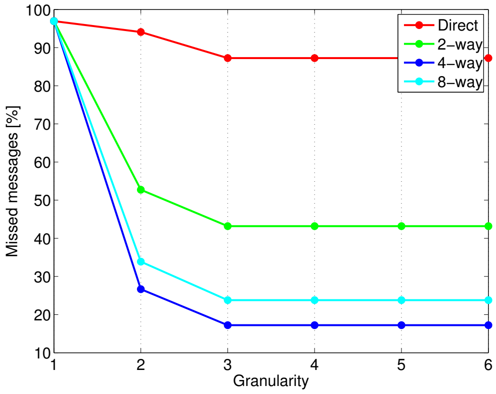

Figure 10: Effect of associativity on predictor performance with start digit = 4.

We have also shown that these techniques work for systems spanning multiple chips with longer time of flight between tile and switch. The replacement of separate electonic NoC and off-chip networks with a single photonic network has the potential to reduce both latency and energy consumption in multiple socket servers and could enable efficient larger shared memory systems with increased sockets per card or spanning multiple cards or racks. Nor is there significant power penalty in mutliple chip networks of this kind. Interfaces between on-chip nanophotonic silicon waveguides and larger chip-to-chip polymer waveguides or fiber have been demonstrated with < 0.5 dB loss [26], while polymer and fibre have considably lower transmission losses than silicon waveguides. Employing a separate photonic switch chip as shown in Figure 1b enables a wider range of switching technologies to be considered including semiconductor optical amplifiers (SOA) which can further reduce the processor chip power disipation [12] while retaining silicon photonic elements for the transmitters and receivers which benefit from tight integration with the processing tiles. However, for multiple chip networks, synchronisation latency (see Figure 2) becomes an important issue as the transmitter and receiver do not share the same clock and the latency savings from the prediction algorithms could be negated by the preamble required to recovery the clock at the receiver. Source synchronous wavelength striped photonic links have been demonstrated operating at up to 4 Gb/s [27] and due to the fundamentally lower delay variation in photonic compared with electronic links [28] may also work at higher bit rates.

As well as the latency and energy consumption benefits, the larger shared memory systems with photonic interconnect resulting from this work could promote more efficient programming of emerging applications in big data analysis, media streaming and other large scale data centre operations.

7. ACKNOWLEDGMENTS

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).

8. REFERENCES

[1] A. Shacham *et al.*, "Building ultralow-latency interconnection networks using photonic integration," *IEEE Micro*, vol. 27, no. 4, 2007.

[2] D. Vantrease *et al.*, "Corona: System implications of emerging nanophotonic technology," in *Int. Symp. on Comput. Archit.*, 2008.

[3] A. Krishnamoorthy *et al.*, "Computer systems based on silicon photonic interconnects," *Proc. of the IEEE*, vol. 97, no. 7, 2009.

[4] Y. Pan *et al.*, "Firefly: Illuminating future network-on-chip with nanophotonics," in *Int. Symp. on Comput. Archit.*, 2009.

[5] G. Hendry *et al.*, "Analysis of photonic networks for a chip multiprocessor using scientific applications," in *Int. Symp. on Networks-on-Chip*, 2009.

[6] G. Hendry *et al.*, "Time-division-multiplexed arbitration in silicon nanophotonic networks-on-chip for high-performance chip multiprocessors," *Journal of Parallel and Distributed Comput.*, vol. 71, no. 5, 2011.

[7] D. A. B. Miller, "Device Requirements for Optical Interconnects to Silicon Chips," *Proceedings of the IEEE*, vol. 97, no. 7, 2009.

[8] J. Shin *et al.*, "A 40 nm 16-core 128-thread SPARC soc processor," *IEEE Journal of Solid-State Circuits*, vol. 46, no. 1, 2011.

[9] D. Wentzlaff and other, "On-Chip Interconnection Architecture of the Tile Processor," *Micro, IEEE*, vol. 27, no. 5, 2007.

[10] M. Madarbux, A. Van Laer, and P. Watts, "Low latency scheduling algorithm for shared memory communications over optical networks," in *Symp. on High-Performance Interconnects*, 2013.

[11] I. Iliadis and C. Minkenberg, "Performance of a speculative transmission scheme for scheduling-latency reduction," *IEEE/ACM Trans. on Networking*, vol. 16, no. 1, 2008.

[12] P. Watts *et al.*, "Energy implications of photonic networks with speculative transmission," *IEEE/OSA Jour. of Opt. Comm. and Netw.*, vol. 4, no. 6, 2012.

[13] K. J. Barker *et al.*, "On the Feasibility of Optical Circuit Switching for High Performance Computing Systems," in *Proceedings of the ACM/IEEE Supercomputing Conference*, 2005.

[14] P. Watts *et al.*, "Requirements of low power photonic networks for distributed shared memory computers," in *Opt. Fib. Comm. Conf.*, 2011.

[15] M. Acacio *et al.*, "Owner prediction for accelerating cache-to-cache transfer misses in a cc-numa architecture," in *Supercomputing, ACM/IEEE 2002 Conference*, 2002.

[16] S. Kaxiras and C. Young, "Coherence communication prediction in shared-memory multiprocessors," in *Int. Symp. on High-Performance Computer Architecture*, 2000.

[17] M. Martin *et al.*, "Using destination-set prediction to improve the latency/bandwidth tradeoff in shared-memory multiprocessors," in *Int. Symp. on Comput. Archit.*, 2003.

[18] C. Adi *et al.*, "An efficient path setup for a photonic network-on-chip," in *Int. Conf. on Networking and Computing*, nov. 2010.

[19] U. Ogras and R. Marculescu, "Prediction-based flow control for network-on-chip traffic," in *Design Automation Conference*, 2006.

[20] A. Poon *et al.*, "Cascaded Microresonator-Based Matrix Switch for Silicon On-Chip Optical Interconnection," *Proc. of the IEEE*, vol. 97, no. 7, 2009.

[21] N. Binkert *et al.*, "The gem5 simulator," *SIGARCH Comput. Archit. News*, vol. 39, no. 2, 2011.

[22] C. Bienia *et al.*, "The PARSEC benchmark suite: Characterization and architectural implications," tech. rep., Princeton University, 2008.

[23] J. L. Hennessy and D. A. Patterson, *Computer Architecture, A Quantitative Approach.* Morgan Kaufmann, 4th ed., 2007.

[24] M. Bhadauria, V. Weaver, and S. McKee, "Understanding PARSEC performance on contemporary CMPs," in *Int. Symp. on Workload Characterization*, 2009.

[25] A. Van Laer, T. Jones, and P. M. Watts, "Full system simulation of optically interconnected chip multiprocessors using gem5," in *Opt. Fib. Comm. Conf.*, 2013.

[26] V. R. Almeida, R. R. Panepucci, and M. Lipson, "Nanotaper for compact mode conversion," *Optics Letters*, vol. 28, no. 15, 2003.

[27] C. Gray *et al.*, "Test electronics for a multi-gbps optical packet switching network," in *Electronics Packaging Technology Conference*, dec. 2006.

[28] G. Q. Chen *et al.*, "Predictions of CMOS compatible on-chip optical interconnect," in *Integration, the VLSI journal*, vol. 40, 2007.