

# Selecting a Timing Regime for On-Chip Networks

Robert Mullins, Jeong-Gun Lee and Simon Moore

Computer Laboratory, University of Cambridge

William Gates Building, JJ Thomson Avenue, Cambridge CB3 0FD, UK

*Robert.Mullins@cl.cam.ac.uk*

**Abstract**—Either the presence or the absence of a global clock may be exploited in the design of an on-chip network. In the synchronous case, each router is able to consider a snapshot of global state and build a routing schedule with the knowledge that the system will proceed in a deterministic fashion. This enables a high degree of speculation to be supported, with the ability to make accurate predictions and handle any mispredictions at a low cost. In contrast, a clock-less or asynchronous approach distributes control decisions relying where possible only on local state to maximise fine-grain concurrency. Here it is the ability to generate and react to local control signals quickly that must be exploited rather than a reliance on global state. In this paper we compare and contrast the design of two on-chip networks: one exploiting delay-insensitive asynchronous control and the other implemented in a traditional synchronous design style.

## I. INTRODUCTION

Communication is becoming increasingly important in the design of large VLSI systems. There are many reasons for this trend towards communication focused architectures:

- **Power:** An increasingly large proportion of the total system power is dissipated by the interconnect. Large energy savings are possible by funnelling much of a system's longer distance communications onto a network consisting of highly optimised low-power communication links. At a higher level it is also believed that on-chip networks will fulfil a key role in minimising system power consumption by providing increased freedom to manage and schedule on-chip resources.
- **Performance:** To continue to achieve performance improvements at historical rates will require a move away from increasingly complex fine-grain techniques for exposing and exploiting parallelism. In many cases such techniques are already reaching their performance limits. Returns are now often mitigated by the limited scaling of interconnect performance, power budgets and the need to minimise design time. Performance will increasingly be sought by exploiting coarser grained parallelism provided by a larger number of functional blocks exploiting a shared communication infrastructure.
- **Complexity:** Design and verification effort is another reason to limit the size and complexity of individual system components. Complex signal integrity and timing verification of long interconnects is simplified by the repeated use of a small number of network channel designs, each of which need only be verified once.
- **Reuse:** A communication-centric design approach simplifies the reuse of functional blocks and also allows

the construction of highly reconfigurable and flexible systems. Multi-use or platform systems will become increasingly important as rising mask and design costs must be amortised over larger unit volumes.

In theory the overhead in terms of additional area and latency imposed by networking communications could be low. Networking in its simplest form only involves augmenting existing repeater stages with the ability to switch and buffer data. In practice, the need to support numerous traffic types and necessary architectural enhancements to boost network throughput make designs significantly more complex. Many proposed network router designs represent a significant latency overhead, increasing long distance communication times by a factor of 5-10 even when the network is lightly loaded. A key aim of this work is to investigate how best to realise high-throughput chip-wide networks while maintaining low communication latencies. This paper explores the impact of the choice of timing regime on network architecture and performance.

## II. VIRTUAL CHANNEL ROUTERS

One approach to providing an on-chip communication infrastructure is to implement a packet-switched network employing virtual channel flow control [1]. Virtual channel routers extend wormhole routing techniques to improve throughput and latency. Each physical channel between routers is now shared between a number of virtual channels, where each virtual channel is provided with its own input buffer at the receiving router. The addition of virtual channels improves performance by allowing flits<sup>1</sup> from multiple packets to be sent in an interleaved manner across a single physical channel. This overcomes the limitation of wormhole routing where the inability to interleave flits prevents blocked packets from being bypassed.

A packet progresses in the network by requesting a new virtual channel on the required output at each router node. The packet's individual flits then arbitrate for access to the physical output channel on a cycle-by-cycle basis, competing with flits from other packets requiring the same output. A block diagram of the buffering and switching resources required in such a router are shown in Figure 1.

A key goal in designing such a router is to implement the control logic without adding significantly to the cycle-time determined by the datapath.

<sup>1</sup>A packet is composed of a number of flits (flow-control digits)

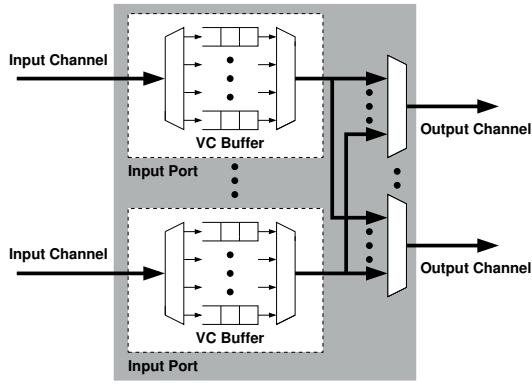


Fig. 1. Virtual Channel Router Datapath

### III. SYNCHRONOUS ROUTERS

A simple approach to implementing control in the case of a synchronous router is to pipeline the various control and datapath operations. For example, a router pipeline could consist of the pipeline stages illustrated in Figure 2(a). Such an approach allows the network to operate at a high clock frequency but unfortunately extends communication latencies and increases buffer credit round trip times. The time taken to traverse the router's datapath is now a small proportion of the total latency of the router pipeline. Some proposed schemes maintain employing a deep pipeline is beneficial and exploit up to 7 pipeline stages per router [2].

The depth of the pipeline may be reduced to a single stage by exploiting various forms of speculation. This reduction in the number of pipeline stages may be achieved without a significant increase in cycle time. Peh and Dally [3] describe how virtual channel (VC) allocation and switch allocation may be performed concurrently (Figure 2(b)). The technique speculates that a waiting packet will be successful in acquiring

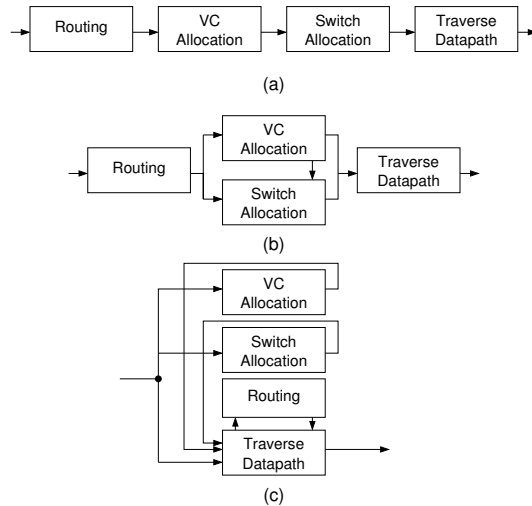


Fig. 2. Virtual Channel Router Pipelines

a VC, this enables it to arbitrate for access to an output before the VC is actually assigned. Speculative requests such as this are only considered if none of the requesting packets already hold virtual-channels.

Further speculation [4] and the use of lookahead routing [5] enables the router pipeline to be reduced to a single stage (see Figure 2(c)). The control logic now produces datapath control signals one cycle in advance of their use. If there is a shortage of buffered flits and an output cannot be scheduled with certainty for the next clock cycle, control signals may be set speculatively. In this case, any flit arriving on the next clock cycle is able to utilise the output port. The prediction made is that only one flit will arrive requiring the output port and that arbitration will be unnecessary. If this prediction is subsequently determined to be overly optimistic the sending of flits may be aborted for the current clock cycle. At the end of the cycle the allocation logic will have determined a valid schedule for sending the flits on subsequent clock cycles.

#### A. Implementation

An implementation of a 16 node 64-bit on-chip network has been completed in a 180nm process. Operating at 250MHz, each of the 48 channels achieve an inter-router data transfer rate of 16Gbit/s. In the best case, each network router hop is completed in one clock cycle. The design operates at a clock cycle time of around 35 FO4 delays. Each router supports five input ports and five output ports. Four virtual channels per physical link are provided and each virtual channel input buffer may hold four flits.

One potential drawback of a synchronous implementation are the problems associated with distributing a low-skew global clock. In addition to a traditional H-tree design a novel solution to the generation and distribution of a global clock was investigated. The Distributed Clock Generator (DCG) [6] is a closed-loop system which is able to maintain a very low clock skew while requiring far less power than a H-tree.

Further cycle-time improvements could be made to the synchronous control scheme by pipelining the generation of the control signals backwards over more clock cycles. Performance benefits of such an approach would be limited by an increase in speculation.

### IV. CLOCK-LESS ROUTERS

A potential pitfall in attempting an asynchronous design is to fail to fully exploit the benefits of distributing control. For this reason the synchronous control architecture outlined previously provides a poor starting point for developing an asynchronous design. The synchronous approach relies on utilising snapshots of global state that are costly to emulate when the clock is removed. Furthermore, the overheads associated with use of speculation are potentially higher when control is distributed.

Our first step in developing an asynchronous router is to decouple the major functions of the router. This distributes control and minimises the existence of synchronisations that may force the design to operate in a pseudo-synchronous

manner. An outline of the proposed scheme is presented in Figure 3.

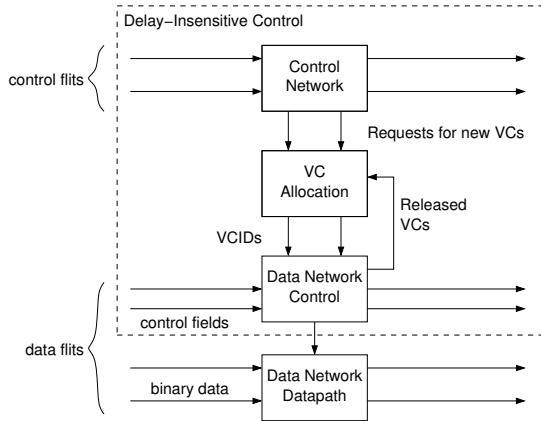


Fig. 3. An Asynchronous Router Architecture

Data is now steered through the latches and multiplexers of the binary encoded datapath by control circuits implemented in a delay-insensitive design style. This approach relies on encoding data in such a way that its presence or absence may be detected (rather than assumed in the synchronous case) [7].

The control and data networks of the asynchronous router operate in a highly decoupled fashion. The role of the control network is to transport single flit packets which setup the route and initiate VC allocation for the data flits at each router node. The control network implementation itself is very simple, requiring no virtual channels and having to contend with a lighter traffic load than the data network. For this reason the control network is able to run ahead of the data network, preventing where possible VC allocation adding to packet communication latencies. This notion of a fast control network can also be exploited in a synchronous environment [8]. Performance is also aided by exploiting cases where a request to send a packet may be initiated before the actual data is available.

The data network control operates on the delay-insensitive encoded control field that accompanies each binary encoded data flit. This control field indicates the arrival of new data and specifies the VC buffer it should be latched in. The control field also identifies the data flit as a head/body flit or a tail flit. The departure of a tail flit enables the VC allocated to the packet to be recycled. A subtlety of the scheme is the need for the order in which VCs are recovered at a particular output channel to be communicated to the router at the other end of the channel. This enables newly assigned VC identifiers (and the output port required by the packet) to be steered to the correct input VC buffer.

At the lowest level the data network control is able to run ahead of the datapath latch controllers when possible to hide control overheads (such as arbitration). Arbitration delays themselves are minimised by using low-latency tree arbiters [9], [10]. The asynchronous router's datapath exploits a deeper pipeline than the synchronous one - with latches

placed between the input ports and output port multiplexers.

### A. Implementation

An implementation of an asynchronous router with the same architectural parameters as that of the fabricated synchronous design is under construction. The design is based on the same standard-cell library with the addition of a small number of custom cells (mutual-exclusion element and C-elements). In order to make the design competitive with the synchronous router a number of optimisations are being explored:

- 1) **Circuit-Level Optimisations** A number of circuit-level optimisations could be employed to boost performance. These include relaxing the strict delay-insensitive circuit style to include a number of simple timing assumptions. The drawback of such an approach is that each timing assumption must be verified post-layout. At the layout-level the performance of a small number of critical control wires could be improved by increasing their width and spacing. Such an approach could enable the control network to run further ahead of the data network or enable data network control fields to arrive at the router ahead of the actual binary data.
- 2) **Arbitration** A major component of the latency and "cycle" time of the asynchronous router are the two tree-arbiters required to gain access to an output port. One technique which could be used to improve performance would be to allow the control flit to make switch arbitration requests (in addition to VC requests). The arbitration results could then be queued to form a switch schedule (for each output port). An additional enhancement would be to enable each output to operate in two modes, the first would allow control flits to make switch arbitration requests and the second would force data flits to make requests when they arrived. The second mode would be used when network traffic is higher to ensure optimal use of available resources.
- 3) **Low-Latency FIFOs** The latency of the router could be reduced by replacing the micropipeline style input FIFOs with parallel FIFO designs. Another possibility is to adapt to the amount of network traffic by dynamically increasing or decreasing the amount of buffering for each VC. Such an approach could be used to minimise latency when network load is low and provide additional buffering for improving throughput at high loads. Such adaptation is possible by configuring a subset of latches to remain permanently transparent.
- 4) **Make Common-Case Fast** Many of the optimisations suggested above rely on the ability of asynchronous circuits to exploit local variations in delay. The router design could be optimised to further exploit such variations *e.g.*, the delays involved in routing flits in a straight-line could be optimised over those involved in a turn, or a particular virtual channel could be implemented to operate faster by skewing the implementation of arbiters and the datapath multiplexers.

## V. RELATED WORK

A number of asynchronous on-chip network router designs have recently been published [11], [12], [13]. The problem of VC allocation in such designs is avoided by assuming the packet utilises a single VC assigned statically at source. The performance impact of such a simplification at the system-level is not investigated. A low-cost asynchronous switched-interconnect architecture constructed from narrow delay-insensitive links is presented in [14]. An asynchronous crossbar designed for networking synchronous blocks operating at different clock frequencies is presented in [15].

A wide-range of synchronous designs have been presented to date. A complete list is not possible here, but some notable contributions include [1], [2], [3], [8], [16], [17].

## VI. CONCLUSIONS

Making a comparison between a synchronous and an asynchronous implementation is not straightforward. Not only is the underlying gate-level design radically different, but architectural changes must be made to exploit the environment each presents. Although a detailed system-level comparison of network performance is beyond the scope of this paper a number of interesting observations may be made.

While it is believed a competitive router and network implementation may be constructed using either approach, an asynchronous scheme may be favoured in a number of cases. One such case is when the system is composed of a number of heterogeneous blocks where link lengths and bandwidth requirements vary widely. In this case an asynchronous design would naturally be able to exploit the range of local delays and ease integration. The concept of a Globally-Asynchronous Locally-Synchronous (GALS) system can be supported independently of the timing regime chosen for the router. Although it can be argued in this case that a fully asynchronous network provides an elegant and potentially higher performance and more flexible solution [15]. It is also believed that if very low latency chip-wide communication is required at lower network loads, asynchronous designs will in theory provide superior results.

A synchronous approach is intuitively best able to exploit a regular network where links lengths are equal and components operate at a fixed clock frequency (or at some rational multiple). This deterministic environment may be exploited to reduce latency and boost network throughput using techniques such as speculation.

## ACKNOWLEDGEMENTS

This work is supported by the Cambridge-MIT Institute and the Ministry of Information and Communication, South Korea.

## REFERENCES

- [1] W. J. Dally and B. Towles, "Route Packets, Not Wires: On-Chip Interconnection Networks," in *Proceedings of the 38th Design Automation Conference (DAC)*, June 2001.
- [2] D. Bertozzi and L. Benini, "Xpipes: A Network-on-Chip Architecture for Gigascale Systems-on-Chip," *IEEE Circuits and Systems Magazine*, vol. 4, 2004.
- [3] L.-S. Peh and W. J. Dally, "A Delay Model and Speculative Architecture for Pipelined Routers," in *International Symposium on High-Performance Computer Architecture*, Jan 2001, pp. 255–266.
- [4] R. D. Mullins, A. F. West, and S. W. Moore, "Low-Latency Virtual-Channel Routers for On-Chip Networks," in *Proceedings of the 31st Annual International Symposium on Computer Architecture (ISCA)*, 2004.
- [5] M. Galles, "Scalable Pipelined Interconnect for Distributed Endpoint Routing: The SGI SPIDER Chip," in *Proceedings of Hot Interconnects Symposium IV*, 1996.
- [6] S. Fairbanks and S. Moore, "Self-timed circuitry for global clocking," in *Proceedings of the 11th IEEE International Symposium on Asynchronous Circuits and Systems*, 2005.
- [7] J. Sparsø and S. Furber, Eds., *Principles of Asynchronous Circuit Design*. Kluwer Academic Publishers, 2001.
- [8] L.-S. Peh and W. J. Dally, "Flit-Reservation Flow Control," in *International Symposium on High-Performance Computer Architecture*, Jan 2000, pp. 73–84.
- [9] M. B. Josephs and J. T. Yantchev, "CMOS design of the tree arbiter element," vol. 4, no. 4, pp. 472–476, Dec. 1996.
- [10] A. Yakovlev, A. Petrov, and L. Lavagno, "A low latency asynchronous arbitration circuit," vol. 2, no. 3, pp. 372–377, Sept. 1994.
- [11] E. Beigne, F. Clermidy, P. Vivet, A. Clouard, and M. Renaudin, "An Asynchronous NOC Architecture Providing Low Latency Service and its Multi-Level Design Framework," in *Proceedings of the 11th IEEE International Symposium on Asynchronous Circuits and Systems*, 2005.
- [12] T. Bjerregaard and J. Sparsø, "A Scheduling Discipline for Latency and Bandwidth Guarantees in Asynchronous Network-on-chip," in *Proceedings of the 11th IEEE International Symposium on Asynchronous Circuits and Systems*, 2005.
- [13] D. Rostislav, V. Vishnyakov, E. Friedman, and R. Ginosar, "An Asynchronous Router for Multiple Service Levels Network on Chip," in *Proceedings of the 11th IEEE International Symposium on Asynchronous Circuits and Systems*, 2005.
- [14] J. Bainbridge and S. B. Furber, "Chain: A delay-insensitive chip area interconnect," *IEEE Micro*, vol. 22, no. 5, pp. 16–23, 2002.
- [15] A. Lines, "Nexus: An Asynchronous Crossbar Interconnect for Synchronous System-on-Chip Designs," in *Proceedings of the 11th Symposium on High Performance Interconnects*, 2003.
- [16] M. B. Taylor, W. Lee, S. P. Amarasinghe, and A. Agarwal, "Scalar Operand Networks," *IEEE Transactions on Parallel and Distributed Systems (Special Issue on On-Chip Networks)*, Feb. 2005.
- [17] J. Dielissen, A. Radulescu, K. Goossens, and E. Rijpkema, "Concepts and Implementation of the Philips Network-on-Chip," in *IP-Based SOC Design*, Grenoble, France, Nov 2003.