Bandwidth Optimization in Asynchronous NoCs by Customizing Link Wire Length

Junbok You
Electrical and Computer Engineering,
University of Utah
jyou@ece.utah.edu

Daniel Gebhardt
School of Computing,
University of Utah
gebhardt@cs.utah.edu

Kenneth S. Stevens
Electrical and Computer Engineering,
University of Utah
kstevens@ece.utah.edu

***Abstract*— The bandwidth requirement for each link on a network-on-chip (NoC) may differ based on topology and traffic properties of the IP cores. Available bandwidth on an asynchronous NoC link will also vary depending on the wire length between sender and receiver. We explore the benefit to NoC performance when this property is used to increase bandwidth on specific links that carry the most traffic of an SoC design. Two methods are used to accomplish this: specifying router locations on the floorplan, and adding pipeline latches on long links. Energy and latency characteristics of an asynchronous NoC are compared to a similarly-designed synchronous NoC. The results indicate that the asynchronous network has lower energy, and link-specific bandwidth optimization has improved the average packet latency. Adding pipeline latches to congested links yields the most improvement. This link-specific optimization is applicable not only to the router and network we present here, but any asynchronous NoC used in a heterogeneous SoC.**

I. Introduction

As more IP cores can be integrated into a System-on-Chip (SoC), core to core communication complexity is increasing and becoming a dominant factor in determining SoC performance. Network-on-Chips (NoCs) are a rising solution to this communication challenge. NoCs are based on common network principles, and share physical resources among multiple concurrent communications. They have started to replace the traditional SoC interconnect structures, shared buses and point-to-point links, which are limited by their scalability in performance, design complexity, and energy-efficiency.

For some SoC designs, the NoC can be specialized for improved performance and energy characteristics [1]. This class of SoC is titled *application-specific*, and is dedicated to a specific task or set of tasks. Its traffic patterns and properties can be known to some extent at design time, such as required average bandwidth (BW) between each specific core. In contrast, a general-purpose chip-multiprocessor (CMP) and its NoC cannot often make such assumptions of traffic between particular cores, as it runs a much larger variety of software.

Asynchronous communication across links and routers is performed using a handshaking protocol rather than a synchronous clock signal. Typically a *request* and *acknowledge* signal, seen in Figure 1, are used to accomplish this. The sender generates the request signal to notify that new data is available. The receiver responds with the acknowledge signal, indicating the data is stored.

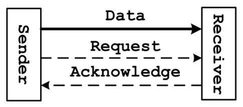


Fig. 1: Typical asynchronous handshake protocol.

Asynchronous design has a number of advantages when designing a NoC. These include a decreased latency in uncongested networks, no global clock distribution energy, zero dynamic power consumption when idle, robustness to variations, and the ability to operate at arbitrary frequencies.

Unfortunately, traditional asynchronous handshake protocols are particularly poorly suited for NoC communication tasks, where significant delay occurs in transmitting a signal between sender and receiver. Two-phase protocols require at least two time-of-flight wire delays per data transfer. This decreases the available bandwidth (ABW) on long links versus a clocked design by up to a factor of two.

Wire propagation delay down a network link dictates the performance of an uncongested asynchronous NoC. In such cases an asynchronous design is faster than a clocked design, which stalls outgoing data until the clock edge occurs. However, if the link is congested, a new packet can be stalled for a much longer time in an asynchronous NoC than in a comparable clocked design due to reduced ABW. Thus asynchronous handshake overhead primarily manifests itself in congested networks. This typically results in asynchronous networks saturating their throughput at a lower offered load than clocked designs.

A new method is presented for mitigating the overhead of asynchronous handshake protocols in NoC designs. This is achieved through creating a custom multi-frequency network. Due to wire delay associated with the propagation of handshake signals, the ABW and frequency of an asynchronous design is dictated by the proximity of the controllers. The frequency and bandwidth of a link will increase by reducing link wire length. Asynchronous design has the ability to operate at arbitrary frequencies. Thus the planned placement of routers as well as the addition of pipeline latches can be employed to customize the ABW of each link in an asynchronous network. This technique is used to reduce the handshake overhead on performance critical links.

This paper evaluates the utility and benefit of customizing the ABW of each individual link in an asynchronous NoC by designing asynchronous and synchronous networks and comparing the designs for performance and energy. The bandwidth of each link in the asynchronous design is adjusted according to the bandwidth requirements of an SoC design specification. This localized bandwidth control is explored through placement of pipeline latches into strategic locations on particular links. The property of controlling per-link bandwidths cannot be easily duplicated in a synchronous NoC and is not performed on this design.

Results show that bandwidth optimization not only mitigates the handshake protocol disadvantage of asynchronous designs, but allows them to operate at a higher offered load than comparable clocked designs.

This paper is organized as follows. Section II gives an overview of related work. The asynchronous and elastic clocked router designs are discussed in Section III. Section IV describes our methodology for simulation. With the results discussed in Section V, we conclude in Section VI.

II. Related Work

This work draws upon two lines of research: asynchronous communication link properties and asynchronous router and NoC designs.

Communication link properties are of critical concern in NoC design. As process technology scales down, wire delay increases relative to gate delay and can have a significant impact on communication performance. Previous work has been developed to create first order energy and ABW models for common communication protocols including 4-phase and 2-phase asynchronous handshake protocols, delay-insensitive encodings, and clocked communications [2]. A new protocol to increase ABW across asynchronous handshake channel with long wires by employing a twin request/acknowledge control scheme has been developed [3].

Several asynchronous NoC designs have been presented mainly in terms of their router architecture, support for Guaranteed Service (GS) and Best Effort (BE), virtual channel (VC) implementation, handshake protocol, and performance evaluation.

MANGO is an asynchronous router which supports both GS and BE packet transfer [4], [5]. Connection-oriented GS transfer is accomplished through VC links, while BE transfer is based on standard connectionless transfer.

CHAIN uses quasi delay-insensitive 1-of-4 encoding [6]. Its BE packets are source-routed with wormhole switching. Particularly, its network fabric is composed of steering blocks and arbiter blocks with separate command and response paths in an irregular network topology.

The QNOC is also an asynchronous router design aiming quality-of-service guarantees using multiple service levels [7]. The 4-phase bundled-data (BD) protocol is used for both the router and links.

ANOC is an asynchronous NoC implemented with quasi delay-insensitive 4-phase protocol designed for the FAUST architecture which is a SoC platform for telecommunications [8], [9]. It uses wormhole switching and 5-port router assembled into a mesh topology.

To the best of our knowledge this work is the first to specifically evaluate how link ABW, as affected by link wire length, determines asynchronous NoC performance. None of the previous publications on router and NoC design have considered this property. A related work presents a link capacity allocation algorithm for application-specific NoCs [10]. Each link is assigned a different ABW according to expected BW requirements. It compares this to a NoC in which all link ABW are identical. However, it does not specifically explain how the ABW is adjusted, nor describe using link length as the parameter. This paper showcases techniques to do this adjustment which should be applicable to other asynchronous NoCs, not only our own.

III. Router Designs

A. Overview

The router designs reported here are intended for efficiency through simplicity. Both the clocked and asynchronous designs employ similar structure and architecture. To achieve this, we chose somewhat unconventional parameters including: a) simple source-routing, b) single-flit packet and c) simple high throughput and low latency network router design. Each switch directs a flit to one of two output ports. With bi-directional channels, this results in a three-ported "T" router. The packet format consists of a single flit containing source-routing bits in parallel, on separate wires, with the data bits. The packet is switched through a simple demultiplexer controlled by the most-significant routing bit. The address bits are simply rotated, or swizzled, for the output packet to place the next routing bit in the most significant position.

The router is composed of three switch and three merge modules, as shown in Fig. 2. Each switch and merge module has one set of latches providing 1-flit buffers on each input and output port. The switch module steers incoming data to one of the other two outgoing ports. The merge module arbitrates between two input requests to an output port.

B. Asynchronous Router

The asynchronous router was designed to use, internally, a bundled-data 4-phase protocol to facilitate arbitration, while a BD 2-phase protocol is used on the link between routers, as this has half the wire transitions as 4-phase, and thus half the total time-of-flight link delay per transaction.

The design of the switch module is shown in Fig. 3a. A 2-to-4 phase converter is implemented on the input control channel (signals *lr* and *la*). This handshakes with a BD 4-phase burst-mode asynchronous linear controller to pipeline the data. The linear controller has the same specification and timing assumptions as the one used in [10]. The output request is steered to one of two channels (*rr1* or *rr2*) based on the most significant routing bit with a demultiplexer.

The merge module is composed of the arbitration circuit, *ar*, and merge controller shown in Fig. 3b. The arbitration circuit serializes requests to the shared output channel, and

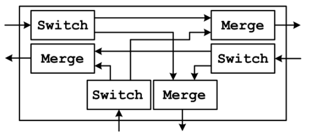
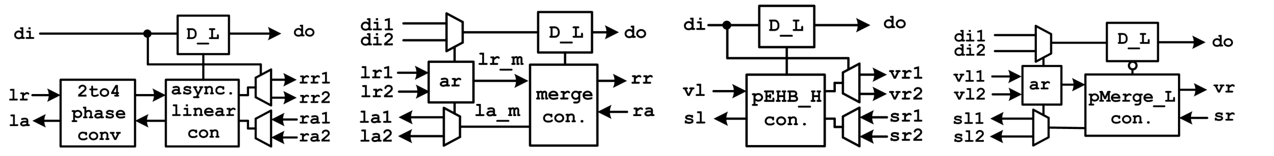


Fig. 2: Architecture of a 3-port asynchronous router.



(a) Async. Switch (b) Async. Merge (c) pSELF Switch (d) pSELF Merge

Fig. 3: Block diagrams of the asynchronous switch and merge, and the pSELF switch and merge. D_L:Data Latch.

controls a MUX that selects which input data to store in the output latch. Each arbitration circuit transaction requests a data transfer via the 4-phase handshake signal *lr_m*. This request passes through the merge controller to generate the 2-phase network link handshake on signals *rr* and *ra*, as well as store the data in a latch. The arbitration circuit grants access to the first-to-arrive request signal. A more complete description of the asynchronous router design is presented in [11].

C. Synchronous Router

Latency insensitive protocols (LIP) are an adoption of asynchronous handshaking for a clocked system, and thus operate with a similar flow control method as the asynchronous protocols [12]. The similarity results in analogous LIP router architectures that use handshake signals, as well as a clock, for timing and sequencing. This allows a generally fair comparison of the effect of the communication links on NoC performance by minimizing other factors which may come from the flow control and router designs.

A specific latency insensitive protocol, called pSELF (phase Synchronous ELastic Flow), was employed here for the synchronous router [13]. This protocol is similar to the SELF protocol [14].

The pSELF switch and merge modules are shown in Fig. 3c and 3d. Their operation is basically identical to that of their asynchronous counterparts. The arbitration circuit of the pSELF merge uses a round-robin scheme when two valid inputs (*vl1* and *vl2*) are contended. The pSELF switch uses a half buffer latch *pEHB_H* active on the high clock phase while the *pMerge_L* latch operates in the low phase of the clock. Clock gating is inherently implemented in pSELF as part of the protocol since the data latch is clocked only when the valid signal (*vl*) is active.

D. Router Design Evaluation

Design results of the two router architectures are summarized in Table I. The pSELF router has better maximum throughput, while the asynchronous router uses less energy and area. We synthesized the router circuits with the static regular V_{th}, Artisan cell library on IBM's 65nm 10sf process using Synopsys Design Compiler, and physically placed and routed designs using Cadence SOC Encounter. Functionality and performance were validated in the designs with ModelSim using back-annotated post-layout delays. Energy per flit, which is consumed when one flit passes a router from an input port to an output port, was measured by HSPICE simulations after parasitic extraction using Mentor Graphics Calibre PEX. A 25% data switching activity factor was applied to the data bits for the energy measurements.

I: Router Design Summary

	Async.	pSELF
Max. Throughput (Gflits/s)	2.12	2.90
Dynamic Energy/flit (pJ)	1.03	1.35
Dynamic Idle Energy/clk (pJ)	0.00	0.30
Area (μm^2)	2828	3761

Dynamic idle energy per clock is the energy consumed by transitions of a gated clock when there is no valid flit transfer. There is no such energy consumption in the asynchronous router.

Fig. 4 reports dynamic energy per flit, including the idle-cycle energy, used to send the same amount of data at various rates. The asynchronous router is compared against the pSELF router operating with a 1 GHz, 2 GHz, and 2.90 GHz clock. As can be seen, the asynchronous router uses the same dynamic energy per flit regardless of the link idle times. However, dynamic energy per flit in clocked routers is sensitive to the ratio of active versus idle cycles. As clock frequency increases and the flit transfer rate decreases, more aggregate energy is consumed by clock gating logic in the idle time. When the flit transfer rate equals the clock frequency of each pSELF router, there is no energy consumed by idle time clocking. The energy per flit of the pSELF router is 1.35 pJ as shown in Table I.

Fig. 5 depicts how asynchronous communication performance degrades with increasing wire length due to handshake control signal propagation delay. The degradation is due to the overhead of the transit time of acknowledgment signal from the receiver to the sender in a 2-phase protocol. On the other hand, the throughput of the synchronous routers is not changed by the link wire length since it is determined only by its clock frequency. However, the pSELF links do have a maximum wire length that can be supported for any given frequency with which they can operate without link pipelining.

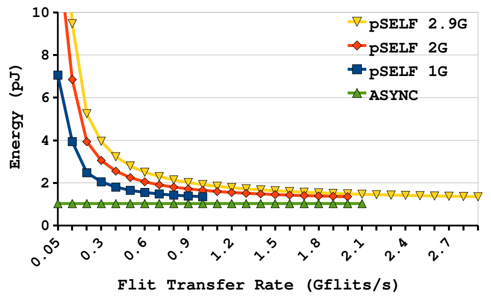


Fig. 4: Router dynamic energy per flit, including idle-cycles, with various flit transfer rates.

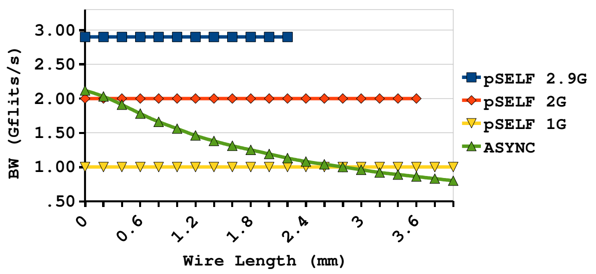


Fig. 5: Link wire length effect on the asynchronous router throughput compared with the pSELF router.

IV. Evaluation Methodology

We evaluate asynchronous and clocked pSELF networks on an abstraction of a MPEG4 decoder design [15]. Communication properties of the design are represented with a *Communication Trace Graph* (CTG), shown in Fig. 6, where nodes are IP cores and weights show required average BW between communicating pairs. Note that the weights have been modified from previous work. The cores were floor planned with the Parquet tool [16].

A custom CAD tool was developed to generate and optimize the topology and router placement for our three ported routers on the floorplan [11]. Its solution is used for both the asynchronous and clocked networks. The generated topology is shown in Fig. 7. This tool reduces the length of high traffic links to save wire energy. For the asynchronous NoC, this artifact also increases ABW on the links that need it most.

A SystemC-based simulator was developed for the asynchronous and clocked networks to model packet latency. The simulations were made as accurate as possible to the physical design by back-annotating the delays extracted from layout into the ModelSim Verilog-SystemC co-simulation. The wire delays for each link are modeled using an interpolation of simulation values [17]. Wire energy per link is estimated with the Orion 2.0 models [18]. The Orion implementation was improved in this work to use more accurate sizing of the buffer driving the first wire segment.

Asynchronous and clocked pSELF networks were implemented using the same topology and router placement. Three different clock frequencies are employed for the clocked pSELF NoC: 1.54 GHz, 2.12 GHz and 2.90 GHz. The 1.54 GHz frequency was selected because it has the same aggregate ABW as the sum of all the links in the asynchronous network. Thus the asynchronous and 1.54 GHz pSELF design have the same average link ABW. The 2.12 GHz pSELF router has the same ABW as the asynchronous router if there were zero wire delay between network nodes (see Fig. 5). The 2.90 GHz is the maximum clock frequency of the pSELF router.

The MPEG4 design was simulated with different BW requirement. The default bandwidth (1×) implements the communication bandwidth values shown in the specification in Fig. 6. Traffic load is increased by multiplying the basely values of each link by the same factor, resulting in three times the load for a 3× network, by five times for 5×, and so on. This gives a comparison at increased traffic loads.

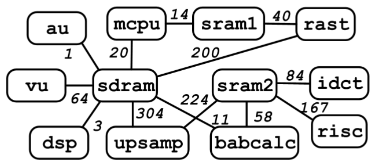


Fig. 6: MPEG4 CTG graph. Edge weights are in MBytes/s.

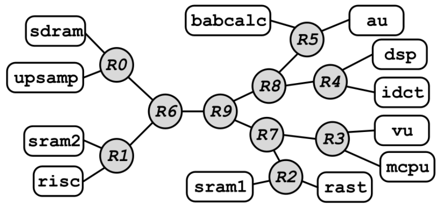


Fig. 7: MPEG4 network topology.

V. Results

In this section results are presented that show the asynchronous NoC provides lower average packet latency and uses less energy than the clocked pSELF NoCs.

A. Link Utilization

Link utilization is measured for all 42 links for the asynchronous, pSELF 1.54G and pSELF 2.12G NoCs. The ABW and utilized load are shown for 12 of these links in Fig. 8 for the 5× offered load simulation. The two pSELF NoCs have identical ABW on all links due to their synchronous nature and global clock frequency. The ABW of each asynchronous link differs based on its individual link wire length determined by the network topology and router placement tool considering traffic load of each link.

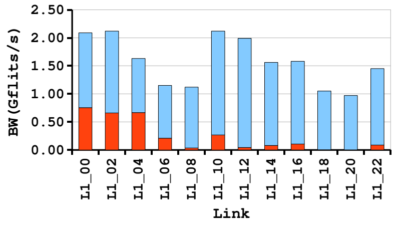
The pSELF 1.54G NoC shows higher link utilization percentage in high traffic load links (L1_00 (48.9%) and L1_02 (42.8%) in Fig. 8b) than the asynchronous NoC. These links will become congested earlier with increasing offered traffic. However, a high link utilization points to less wasted energy from idle clock cycles.

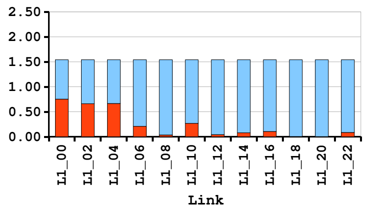
The pSELF 2.12G NoC shows very low link utilization percentages in low traffic load links (L1_08 (1.67%) and L1_20 (0.06%) in Fig. 8c) compared to the asynchronous NoC. Higher operating frequency is beneficial for low packet latency, and it also improves the capability to handle higher traffic load. However, it has more idle cycles on the low traffic links, which wastes considerable energy from idle clocking.

The asynchronous NoC has different ABW of each link, as shown in Fig. 8a. Links carrying more traffic are given more ABW thanks to shorter link wire lengths. For example, L1_00 has ABW of 2.09 Gflits/s and L1_02 has 2.12 Gflits/s. These are similar to the ABW of each link in the pSELF 2.12G NoC. Links with light traffic loads are assigned low link ABW. For example, L1_08 has 1.12 Gflits/s and L1_20 has 0.97 Gflits/s which are significantly less than the ABW of the pSELF 1.54G NoC.

B. Average Latency

Fig. 9 compares the average latency of the asynchronous network and three pSELF networks with varying offered


(a) Async.


(b) pSELF 1.54G

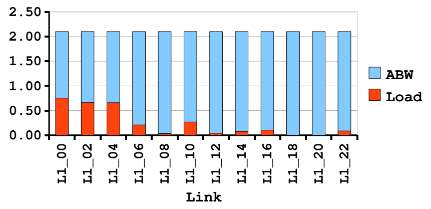

(c) pSELF 2.12G

Fig. 8: Link utilization in the asynchronous, pSELF 1.54G and pSELF 2.12G NoCs in 5× offered traffic load. *ABW* is a available link BW, and *Load* is traffic load of each link labeled on X-axis.

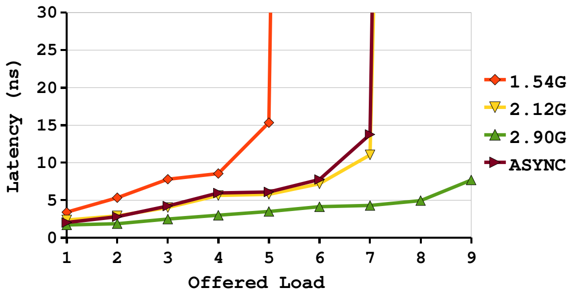


Fig. 9: Average latency comparison between the asynchronous and pSELF networks in various offered loads.

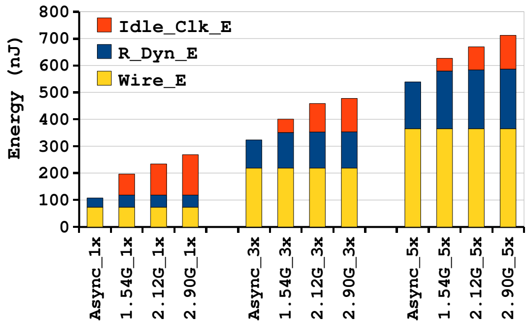


Fig. 10: Energy distribution at 1×, 3× and 5× offered loads.

traffic loads. An increase of latency as offered traffic load rises shows that traffic paths contend for switch and link resources for long periods of time.

The pSELF design clocked at 1.54 GHz has longer latency at a lower traffic loads. Here, packet latency is determined mainly by the clock period since the network is largely uncongested. This is larger at 1.54 GHz than the asynchronous network and the two other higher frequency networks. Furthermore, its saturation point is at 5× load. This occurs because some links have higher utilization (L1_00 and L1_02 in Fig. 8b), and these links become fully congested earlier than other networks which have more ABW in those links.

The asynchronous network and pSELF 2.12G network show almost identical average packet latency. The clock frequency of 2.12 GHz was selected to match with the maximum throughput of the control logic of the asynchronous router. Note that this frequency of operation is only achieved with zero wire delay in the asynchronous network.

The pSELF 2.90G network shows the lowest average latency. This design is not fully congested even at the highest offered load examined, due to the sufficient ABW in all links. However, this advantage in latency comes at a price of the higher energy usage of a faster clock.

C. Energy

Energy usage is reported in Fig. 10 for each network at three offered loads: 1×, 3× and 5×.

The asynchronous NoC energy consists of the routers' dynamic energy (*R_Dyn_E*) and the wire energy (*Wire_E*). The pSELF NoC energy includes another component, the idle clock energy (*Idle_Clk_E*), which is from the cycles in which routers do not switch flits. The router dynamic energy is the total energy used by all 10 routers in the networks. Because of their architectural similarity, the router dynamic energy is very similar between the asynchronous router and pSELF router. Wire energy (Wire_E) is the sum of energy used by the wires composing the links. Each link energy was calculated based on its length and carried traffic volume. The asynchronous and pSELF networks used the same topology and router placement, thus the link wire energy is identical in all networks.

As a consequence, idle clock energy is the primary differentiator for the total NoC energy between networks. The asynchronous network consumes less energy than all other pSELF networks by as much as the idle clock energy of each pSELF network. The portion of the idle-to-total energy increases as the offered load is lowered, and as the clock frequency is increased, both which lead to more idle cycles. Accordingly, the asynchronous network is more energy-efficient compared to the pSELF of high frequency, particularly when offered load onto the network is low. The asynchronous network consumes 54%, 29% and 20% less energy than the pSELF 2.12G (which has the similar average packet latency) in 1×, 3× and 5× offered loads, and 60% less than pSELF 2.90G in 1× offered load.

Note that energy consumption by clock distribution network for the pSELF NoCs is not included in the energy comparisons. The total energy gap between the asynchronous NoC and the pSELF NoCs will increase when clock tree energy is considered.

D. Link Pipelining by Latch Insertion

A pipeline latch stores a flit and provides link-level flow control on the channel. Pipeline latches on asynchronous channels not only buffer flits but also improve ABW on individual links by reducing link wire length between controllers, whereas synchronous channels benefit from only buffering by pipeline latches without any improvement of link BW.

Fig. 11 shows part of the asynchronous network with pipeline latches placed in links selected as the most congested. The numbers in parenthesis are the wire lengths of the corresponding links. Twenty pipeline latches were inserted in eight links out of a total of 42 in the MPEG4 example.

The effects of pipeline latch insertion was studied by comparing the effects on the asynchronous network and one pSELF design. The pSELF 2.12G was selected since it showed very similar average packet latency. Only 8 pipeline latches were added to the pSELF 2.12G design rather than the 20 used in the asynchronous design. The SystemC simulation assumes infinite sending and receiving buffers at the interface of each IP core. Therefore, inserting pipeline latches in the first level of links which are connected directly to IP cores in a clocked design has no beneficial effect, rather, it merely increases packet latency. For this reason, no pipeline latches are inserted in the first level links, L1_04, L1_05, L1_06 and L1_07, in the pSELF 2.12G network. Buffers were inserted into the same internal links, L2_00, L2_01, L2_02 and L2_03, as in the asynchronous network. On the other hand, for asynchronous networks, pipeline latch insertion into the first level of links improves performance because it increases the link BW.

The ABW increase with pipeline latches is shown for eight particular links in Fig. 12. Three pipeline latches were inserted into links L1_04, L1_05, L1_06, and L1_07, distributed evenly across its length. As a result, the link wire length between pipeline latches in L1_04 and L1_05 is reduced to 213 μm from 854 μm, and the link ABW increases to 2.10 Gflits/s from 1.63 Gflits/s. The ABW of L1_06 and L1_07 were changed from 1.15 Gflits/s to 1.91 Gflits/s, as the wire length between controllers is reduced to 528 μm from 2113 μm. Two pipeline latches were inserted into links L2_00, L2_01, L2_03, and L2_04. The link ABW of L2_00 and L2_01 subsequently increased to 2.13 Gflits/s from 1.89 Gflits/s, and the L2_02 and L2_03 improved from 1.99 Gflits/s to 2.13 Gflits/s.

The complexity of an asynchronous pipeline latch is less than that of a router. This results in higher operating frequency of the pipeline latches allowing a higher bandwidth on repeated segments. This makes more than the maximum throughput of the router without wire delay, 2.12 Gflits/s, to be achieved on some wire segments. This is due to the faster response time of the repeaters, which compensate for the wire delay. Thus on some of the repeated segments ABW of 2.13 Gflits/s is achieved.

Fig. 13 compares the average latency of 4 different configurations: non-pipelined asynchronous (*ASYNC*), pipelined asynchronous (*ASYNC_PL*), non-pipelined pSELF 2.12G (*2.12G*) and pipelined pSELF 2.12G (*2.12G_PL*). Fig. 13a and Fig. 13b depict the same simulation results, but they are different in the range of offered load. Fig. 13a is intended to show average latency in lower offered load especially.

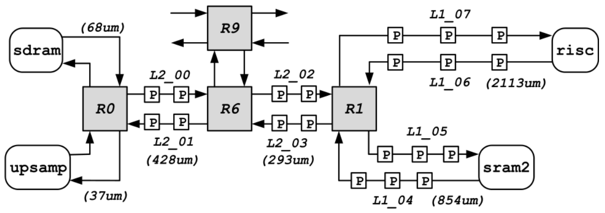


Fig. 11: Pipeline latch placement in the asynchronous network. R:Router, P:Pipeline latch.

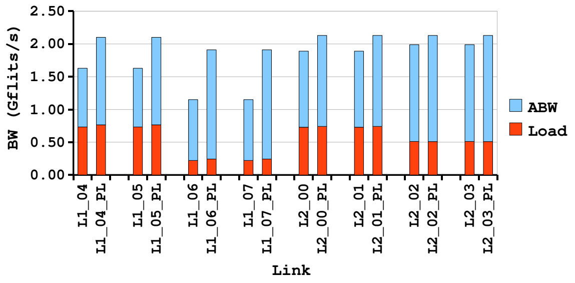


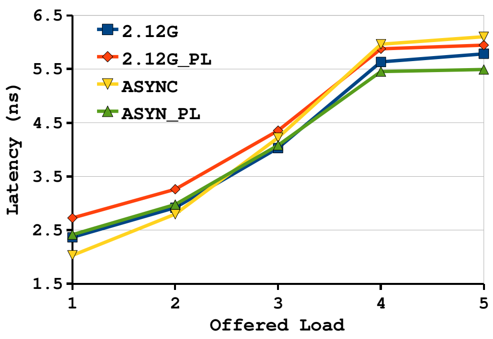
Fig. 12: ABW of non-pipelined and pipelined links of the asynchronous network with 5× offered load.

In comparing *ASYNC_PL* to the non-pipelined (*ASYNC*), the latency improvement due to pipeline latches appears at 3× offered load and beyond. Increasing ABW of congested links relieves network congestion, reducing latency. At 5× offered load the average latency of *ASYNC_PL* is 5.49 ns, which is 10% less than that of *ASYNC* at 6.10 ns. The latency reduction further improves as the offered load increases to 7×; *ASYNC_PL* is 8.76 ns compared to 13.76 ns for *ASYNC*, giving a 36% reduction in average latency with the total NoC energy increase of 6.1% with twenty pipeline latches.

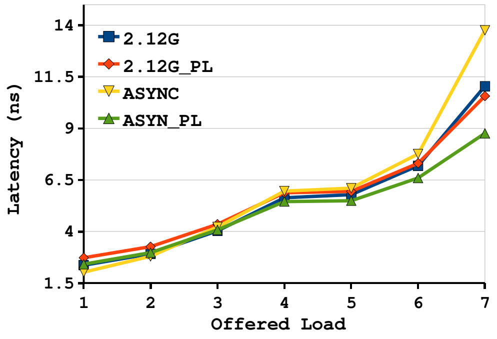
However, at lower offered loads (1× and 2×), *ASYNC_PL* has worse packet latency than *ASYNC*. This is because light traffic loads rarely cause congestion and do not benefit from the additional ABW of pipeline latches. Rather, they just increase latency by adding extra control logic delays to the pure link wire delay.

A similar effect is shown in the pipelined pSELF 2.12G (*2.12G_PL*) network. *2.12G_PL* has worse packet latency than the non-pipelined *2.12G* even in 6× offered load. Inserting two pipeline latches in each link causes each packet latency to increase by one clock cycle (472 ps). These links carry higher percentages of traffic so the detriment to NoC performance is noticeable. However, as the offered load increases to 7×, *2.12G_PL* eventually shows the benefit of pipeline latch insertion with 4.3% less latency than *2.12G* with addition of 3.6% in the total NoC energy.

A key point is that network performance is improved more by inserting pipeline latches into the asynchronous network than the synchronous design. This comes from two factors. First, pipeline latches in the asynchronous link improve ABW as well as add buffering, whereas they serve only for buffering in the synchronous network. Inserting pipeline latches in links reduce handshake cycle time by reduced wire delay as well as the simplified circuitry of the pipeline latch. Second, the asynchronous pipeline latch has much smaller forward latency (120 ps) overhead than the pSELF pipeline latch which has a latency restricted by its clock period.



(a)



(b)

Fig. 13: Average latency comparison between the non-pipelined and pipelined networks in various offered loads.

VI. Conclusion

This paper investigates the benefit of bandwidth optimization in the design of an asynchronous network-on-chip (NoC). First, a tool was developed to optimize NoC bandwidth and energy through topology and router placement [11]. This placement minimizes the wire lengths and router hops for high bandwidth network links. A second optimization is performed on long links that require further bandwidth improvement by adding pipeline latches. This design process creates a multi-frequency bandwidth optimized asynchronous NoC. This design is compared to a clocked NoC applying the same optimizations, but without applying multi-frequency design.

The designs are compared for performance and power. The results show that exploiting the natural multi-frequency nature of asynchronous designs results in substantial improvements. The topology and placement optimizations create an asynchronous design that has an average link bandwidth of 1.54 Gflits/s. Compared to the 1.54 GHz clocked design, the asynchronous design has 46% less average packet latency and 19% less energy consumption at 3× offered load. The asynchronous design performs similarly to the clocked network with a link bandwidth of 2.12 Gflits/s, but demands 29% less energy at 3× offered load. Adding pipeline latches to the clocked design does not have substantial beneficial effect. However, for the asynchronous design this optimization significantly improves performance when the network is highly congested. At 7× offered load, the pipeline latches resulting in a 35% reduction in average packet latency for only 6.1% more energy consumption.

VII. Acknowledgments

This research is supported by the National Science Foundation under grant CCF-0702539 and Semiconductor Research Corporation under task 1817.001. We would like to thank ARM and IBM for providing the 65nm library cells and process technology.

References

[1] L. B. Giovanni De Micheli, *Networks on Chips*. Morgan Kaufmann, 2006.

[2] K. S. Stevens, P. Golani, and P. A. Beerel, "Energy and Performance Models for Synchronous and Asynchronous Communication," *IEEE Trans. on VLSI Systems*, 2010.

[3] R. Ho, J. Gainsley, and R. Drost, "Long wires and asynchronous control," in *The 10th IEEE International Symposium on Asynchronous Circuits and Systems*, 2004, pp. 240–249.

[4] T. Bjerregaard and J. Sparsø, "Implementation of guaranteed services in the MANGO clockless network-on-chip," *IEE Proceedings: Computing and Digital Techniques*, vol. 153, no. 4, pp. 217–229, 2006.

[5] ——, "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, pp. 34–43.

[6] W. J. Bainbridge and S. B. Furber, "CHAIN: A Delay Insensitive CHip Area INterconnect," *IEEE Micro special issue on Design and Test of System on Chip*, vol. 142, No.4., pp. 16–23, Sept. 2002.

[7] R. Dobkin, R. Ginosar, and A. Kolodny, "QNOC Asynchronous router," *VLSI Journal*, vol. 42, pp. 103–115, Feb. 2009.

[8] P. Maurine, J. Rigaud, F. Bouesse, G. Sicard, and M. Renaudin, "Static Implementation of QDI Asynchronous Primitives," in *13th International Workshop on Power and Timing Modeling, Optimization and Simulation (PATMOS2003)*, Sept 2003, pp. 181–191.

[9] I. Miro-Panades, F. Clermidy, P. Vivet, and A. Greiner, "Physical Implementation of the DSPIN Network-on-Chip in the FAUST Architecture," in *NOCS '08: Proceedings of the Second ACM/IEEE International Symposium on Networks-on-Chip*. Washington, DC, USA: IEEE Computer Society, 2008, pp. 139–148.

[10] K. S. Stevens, R. Ginosar, and S. Rotem, "Relative Timing," *IEEE Trans. on Very Large Scale Integration (VLSI) Systems*, vol. 11, no. 1, pp. 129–140, 2003.

[11] D. Gebhardt, J. You, and K. S. Stevens, "Comparing Energy and Latency of Asynchronous and Synchronous NoCs for Embedded SoCs," in *4th IEEE International Symposium on Network-on-Chips*, May 2010, pp. 115–122.

[12] L. P. Carloni, K. L. McMillan, and A. L. Sangiovanni-Vincentelli, "Theory of latency-insensitive design," *IEEE Trans. on Computer aided design of integrated circuits and systems*, vol. 20, pp. 1059–1076, 2001.

[13] J. You, Y. Xu, H. Han, and K. S. Stevens, "Performance Evaluation of Elastic GALS Interfaces and Network Fabric," *Electron. Notes Theor. Comput. Sci.*, vol. 200, no. 1, pp. 17–32, 2008.

[14] J. Cortadella, M. Kishinevsky, and B. Grundmann, "Synthesis of synchronous elastic architectures," in *Proceedings of the Digital Automation Conference (DAC06)*. IEEE, July 2006, pp. 657–662.

[15] E. B. V. D. Tol and E. G. T. Jaspers, "Mapping of MPEG-4 decoding on a flexible architecture platform," in *Media Processors*, 2002, pp. 1–13.

[16] S. Adya and I. Markov, "Fixed-outline floorplanning: Enabling hierarchical design," *IEEE Trans. on VLSI*, vol. 11, no. 6, pp. 1120–1135, Dec. 2003.

[17] L. Carloni, A. Kahng, S. Muddu, A. Pinto, K. Samadi, and P. Sharma, "Accurate predictive interconnect modeling for system-level design," *IEEE Trans. on VLSI Systems*, vol. 18, no. 4, pp. 679 –684, apr. 2010.

[18] A. Kahng, B. Li, L.-S. Peh, and K. Samadi, "ORION 2.0: A Fast and Accurate NoC Power and Area Model for Early-Stage Design Space Exploration," in *DATE*, April 2009, pp. 423–428.