DDRNoC: Dual Data-Rate Network-on-Chip

AHSEN EJAZ, VASSILIOS PAPAEFSTATHIOU, and IOANNIS SOURDIS

Chalmers University of Technology, Gothenburg, Sweden

{*ahsen,vaspap,sourdis*}*@chalmers.se*

Abstract

This paper introduces DDRNoC, an on-chip interconnection network able to route packets at Dual Data Rate. The cycle time of current 2D-mesh Network-on-Chip routers is limited by their control as opposed to the datapath (switch and link traversal) which exhibits significant slack. DDRNoC capitalizes on this observation allowing two flits per cycle to share the same datapath. Thereby, DDRNoC achieves higher throughput than a Single Data Rate (SDR) network. Alternatively, using lower voltage circuits, the above slack can be exploited to reduce power consumption while matching the SDR network throughput. In addition, DDRNoC exhibits reduced clock distribution power, improving energy efficiency, as it needs a slower clock than a SDR network that routes packets at the same rate. Post place and route results in 28 nm technology show that, compared to an iso-voltage (1.1V) SDR network, DDRNoC improves throughput proportionally to the SDR datapath slack. Moreover, a low-voltage (0.95V) DDRNoC implementation converts that slack to power reduction offering the 1.1V SDR throughput at a substantially lower energy cost.

1 Introduction

Chip Multiprocessors (CMP) are one of the most promising solutions for supporting the continuous need for single chip performance improvement. Shrinking transistor geometries still allow more cores to be integrated on a die. However, power constraints prevent chips from fully utilizing all these cores at their maximum performance potential [1]. The on-chip interconnection network is a critical component for power efficiency and performance [2] since roughly a sixth to a quarter of the available chip power budget goes to its interconnects [3, 4, 5, 6]. As a consequence, the design of high-performance and low-power Networks-on-Chip (NoC) is essential for many-core scaling.

Many existing techniques focus on lowering packet latency by modifying the network topology [7, 8], the router architecture [9, 10, 11] or the routing algorithm [12]. Others attempt to improve network throughput employing new allocation techniques [13], wider datapaths, richer network topologies [14], or multiple subnetworks [15, 16, 17, 18]. Packet latency is important for the performance of workloads that exhibit small transfers at low loads. However, more demanding workloads require higher throughput, push the network close to its saturation point, and are more representative of systems that run concurrent scale-out applications [19, 20]. High throughput increases network activity and requires additional power, therefore the energy efficiency of such networks becomes increasingly critical.

In this work, we focus on increasing NoC throughput and improving energy efficiency. We observe that the critical path of existing 2D-mesh routers with VCs is in the control logic. As shown in literature and confirmed by our experiments, the datapath of a typical 3-stage router[1],

[1] In our ASIC place-and-route experiments we consider a 3-stage router (VA/SA, ST, LT), as described in [21], with 4 virtual channels, 5 flit registers per VC and 128-bit wide datapath.

could be 45% faster than its control (allocation) [22, 23, 13]. Moreover, ShortPath, the current fastest published NoC router architecture[2] adds to the Switch Traversal (ST) the delay of half a Switch Allocator and credit-check, which define its critical path [11]. In this case, ST and link traversal (LT) could be clocked at about 25% higher frequency than the ShortPath router. We leverage on this observation and exploit this slack to dynamically boost router's datapath utilization.

We propose DDRNoC, an on-chip interconnection network composed of routers the datapath of which can operate at Dual Data Rate (DDR). Thereby, the rate at which packets are routed (one flit every half a cycle), is limited only by the ST and LT delays rather than the control. DDR mode is enabled for a datapath stage (ST or LT) when more than one packets compete for it. Otherwise, ST and LT operate at DDR when free to route two flits of a single packet. In all other cases, the respective stage operates in Single Data Rate (SDR) mode. On the contrary, the router control has always an entire cycle available. In effect, the datapath can be used twice in a cycle and the control should take (allocation) decisions for up-to two slots per cycle (the two halves of the cycle). This makes the DDRNoC allocation more challenging than that of an SDR router.

The DDRNoC improves throughput over a SDR network by routing packets at the rate determined by the datapath stages (ST, LT) rather than the control. Packet latency is also improved at higher injection rates due to lower contention. Alternatively, a DDRNoC can reduce power consumption using lower voltage circuits, while still matching the SDR network throughput. DDRNoC would then suffer higher latency, as it would be the case in any conventional NoC with reduced voltage and frequency. In all cases, DDRNoC requires slower clock than an SDR network that routes packets at the same rate. Clock distribution power accounts for 20% of total power in our implementations; in literature, that percentage can be as high as 28-33% [24, 25]. So, using a slower clock, DDRNoC improves energy efficiency.

In general, DDRNoC trades power for throughput without adding significant packet latency overheads or trades latency for power savings without compromising throughput. Although network latency may be critical for the overall performance of various systems and application domains (e.g. embedded Systems-on-Chip), there exist many applications, i.e. in High-Throughput Computing (HTC), where that may not be the case. On the contrary, network throughput, the fraction of the power budget devoted to the network, and the Energy-Throughput Ratio (ETR) may be more important in these cases (e.g. concurrent scale-out applications) [20, 26]. For instance, Bakhoda et al. showed that for a wide range of applications using a single stage router versus a four stage router, increasing up to 2× the latency, had from zero to 7% and on average 2.3% increase in overall execution time; on the contrary, doubling the network bandwidth achieved a speedup up to 2× and on average 27% [26].

Concisely, the contributions of this paper are the following:

- a new NoC router architecture for switching packets at DDR, improving throughput and energy efficiency.
- a control-forwarding technique that reduces DDRNoC router latency.
- a switch allocator design that makes two sets of decisions for the two halves of a cycle.
- a DDRNoC implementation in 28nm technology showing post place and route performance and energy gains using synthetic and application driven traffic.
- a comparison with current state-of the art NoC architectures using synthetic and application driven traffic.

[2] ShortPath is a 2-stage router (one stage for routing and one for link traversal) which is reduced to a 3+1 stages in case of contention [11]. ShortPath NoC demonstrated a 9.5% improvement in operating frequency compared to the SCORPIO NoC [10].

The rest of this paper is organized as follows. Section 2 discusses related work. Section 3 describes the DDRNoC architecture. Section 4 presents the implementation and experimental setup. Section 5 shows evaluation results and Section 6 compares with related work. Finally, Section 7 summarizes our conclusions.

2 Related Work

Various existing NoC architectures attempt to exploit the imbalance between the router datapath and control aiming to improve performance. A few of them allow some parts of a network to operate at DDR, but none of them offers DDR in the entire datapath.

Time stealing (or retiming), currently supported by CAD tools, can be used to balance the pipeline of a router. Mishra et al. used time stealing to boost routers performance [27]. In doing so, a router would have a clock period equal to the average of all (3) router stages with some additional delay coming from the additional latches required. On the contrary, DDRNoC throughput is determined by the slowest datapath stage (ST, LT). Indeed, Mishra et al. improve their baseline operating frequency and throughput by 25%, while DDRNoC improves throughput over the same baseline by up to 45% as shown in the comparison Section 6.

Hoskote et al. observed that the crossbar of a 5.1 GHz 5-stage NoC router can be double-pumped [4]. The DDR crossbar is able to handle a flit every cycle requiring half of the width. Dual-edge triggered flip-flops were used to interleave alternate bits of a flit. Replacing two crossbars used for two lanes by a single DDR crossbar of half the width –hence supporting half the original throughput for the two lanes– resulted in router area and power savings of 34% and 13%, respectively.

Xu et al. built DDR Wave-Pipelined (DWP) links interconnecting asynchronously NoC router ports to reduce the number of link-wires [28]. A router converts the data to be transmitted through a DWP link from SDR to DDR format by sending odd bits on the positive level of the clock and even bits on the negative level of the clock. NMOS transmission gates are used for this purpose. A separate wire is used to transmit a reference clock signal to the downstream router, along with the data. This clock signal is used by the semi-static double-edge-triggered-flip-flops (SDETFFs) located at the input of the downstream router to latch the incoming data at both rising and falling edges of the clock. Since this data is being transmitted in DDR mode, the transmitted clock signal needs to toggle at the rate of data transmission.

Recently, RapidLink used DDR links in a NoC architecture [17] based on the consideration that links can operate at double the frequency of a router. RapidLink improves throughput and latency because of two underlying mechanisms. Firstly, it splits a 4-VC NoC in to two parallel physical subnetworks, of 2 VCs each, which share common links. In effect, RapidLink achieves the throughput of two subnetworks that have separate links at half the link cost. Secondly, consecutive subrouters operate in different clock edges allowing them to perform LT in half a cycle and therefore reduce per hop latency. In Rapidlink, link contention between the two subnetworks is avoided, however, contention within a subnetwork router is not addressed. On the contrary, DDRNoC, allows all parts of a NoC datapath (input VC buffers, input port multiplexer, output multiplexer, and link) to operate at DDR. This enables DDRNoC to route at DDR flits of the same or of different packets (stored in VCs of the same or different ports) rather than only flits of two different subnetworks. RapidLink requires links to have limited length in order to be twice as fast as the routers. In such case, throughput is improved in some traffic patterns by about 65% compared to the baseline. However, longer links, as the ones considered in this work, substantially limit RapidLink performance gain as shown in the comparison Section 6.

This last limitation of RapidLink was addressed by the authors in their recent improved design (henceforth called Rapidlink2) where they proposed to pipeline the network links using

Dual Stream (DS) elastic buffers with support for VCs and consider them as part of VC buffers [18]. Thereby, links are no longer in the critical path. However, according to our experiments the energy cost for transmitting a bit (for 128 bits wide flit) over a link that is pipelined using one or two sets of DS elastic buffers with support for 4 VCs increases versus a non-pipelined one by 1.8× and 2.5×, respectively. This is because a register is needed per VC, along with a 4-to-1 mux and an arbiter for each pipeline stage. In Section 6, we show that a DDRNoC with similar datapath modifications –router split in two subnetworks and replicated, rather than pipelined, links– is able to achieve 10% to 27% better throughput than RapidLink2.

In summary, although in the past DDR has been used in parts of a NoC router, DDRNoC is the first approach that allows flits to be routed at a rate determined entirely by the datapath (switch and the link) rather than by the control. Thereby, throughput is maximized or alternatively the slack can be exploited to reduce power consumption using lower voltage circuits.

3 The DDRNoC Architecture

The Dual Data Rate NoC (DDRNoC) is an on-chip interconnect composed of routers that have a double-pumped datapath. As opposed to a conventional SDR NoC router, the critical path of a DDRNoC router is on its switch and link traversal rather than on the control. This allows packets to be routed at a higher rate increasing network throughput. Without loss of generality, our DDRNoC design considers a 2D-mesh network, with look-ahead XY-routing, composed of routers with virtual-channels and credit-based flow control.

The top-level view of the DDRNoC router is shown in Figure 1. The datapath is composed of two stages: Switch Traversal (ST) and Link Traversal (LT). Each stage is able to handle two flits per cycle, one at the high phase and one at the low phase of the clock. There are three main control blocks in the router: Virtual Channel Allocation (VA), Speculative Switch Allocation (SA) and Next-Next-Route-Computation (N^2RC) which are explained in detail below.

We first describe the datapath of a DDRNoC router, then present its timing, explain the individual control blocks, and finally discuss some of our design decisions.

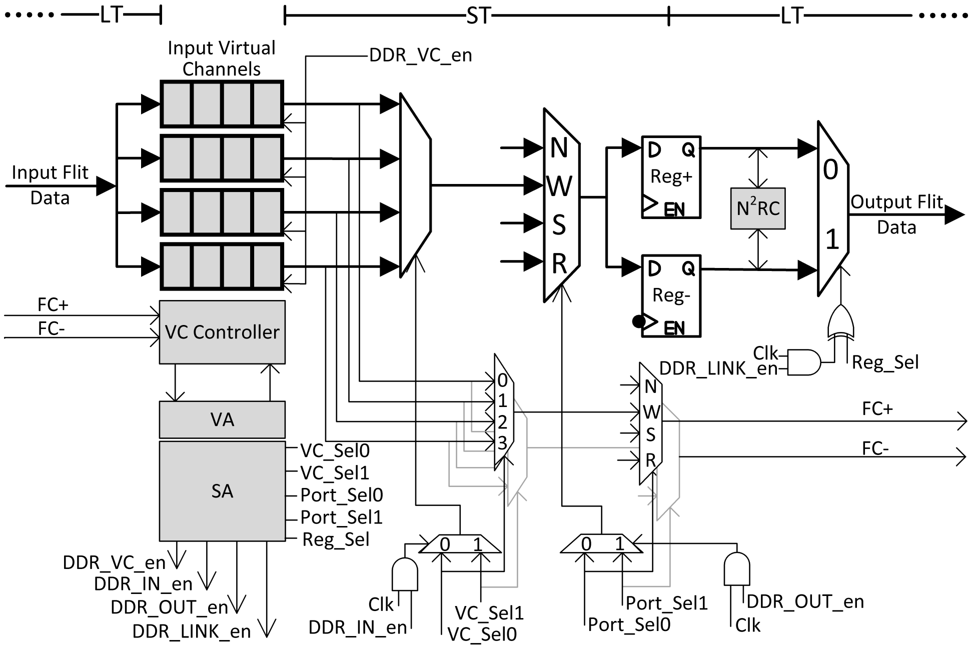


Figure 1: The DDRNoC router architecture.

3.1 Router datapath

The input port of a DDRNoC router is able to receive two flits per cycle, one at each phase of the clock. As shown in Figure 2, the VC buffers are composed of registers that are selectively triggered either at the rising or at the falling edge of the clock to store a flit arriving at the low or the high phase, respectively. The VC buffer control unit along with the Write Enable (WE) Logic, uses the information in the forwarded control signals to selectively propagate a clock pulse to the clock input of the specific register which has to store the incoming flit. This implicitly implements clock gating in the VC buffers as well because triggering clock edges are only propagated to the particular registers which have to store an incoming flit. This mechanism also enables two flits per cycle to be enqueued in a single VC buffer because they are stored in different registers of the buffer. Dequeuing two flits per cycle from a single VC buffer is enabled by the DDR_VC_en signal, which allows two different VC registers to be selected during two consecutive clock phases. Similarly, dequeuing two flits from different VCs within a cycle is enabled by the DDR_IN_en signal, which is used by the input port multiplexer to implement two input arbitration decisions (VC_Sel0 and VC_Sel1). Besides the flit data, the input port receives once every cycle early forwarded control information for the (up to two) received flits ($FC+$, $FC-$). As explained later, this information refers to flits that arrive a cycle later, thus it is called *forwarded control.*

Besides the input port multiplexer, the ST stage includes the output port multiplexer, which applies up-to two output arbitration decisions per cycle ($Port\_Sel0$, $Port\_Sel1$) using the DDR_OUT_en signal. This allows two different input ports to send a flit to the same output port during the high or the low phase of a clock cycle.

A positive edge triggered output register ($Reg+$) and a negative edge triggered one ($Reg-$) are used to store the flits switched in the low and the high phase of a cycle, respectively. Subsequently, a multiplexer selects one of the two registers to send a flit through the link. Using the DDR_LINK_en signal, this multiplexer allows LT of two flits in a cycle. For packet header flits, the N^2RC module computes routing information for two-hops ahead in half a cycle and the result (2-bits) is embedded in the header flit data before LT.

In general, DDR mode is selected in two cases. Firstly, when flits of multiple packets in different VCs compete for the same datapath part (input port multiplexer, output port multiplexer, or link). Secondly, when multiple flits of a single packet are available in a VC buffer

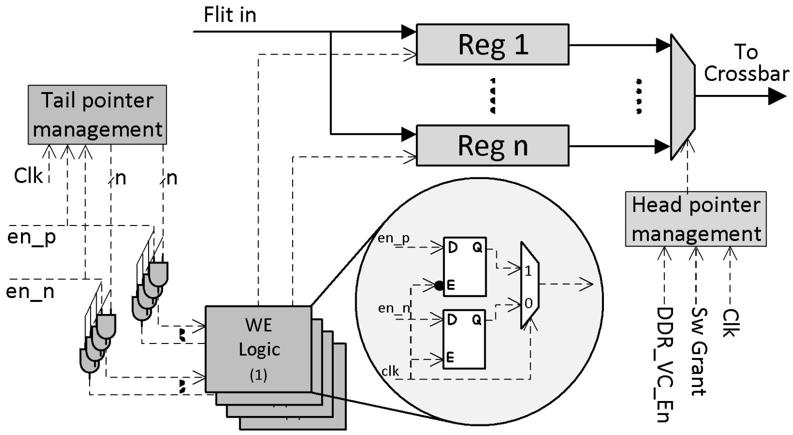


Figure 2: DDRNoC VC buffers. Signals en_n and en_p, derived from forwarded control signals, indicate whether there is an incoming flit at the router input port in the high and low clock phase, respectively. WE Logic uses these signals along with the tail pointer of the VC FIFO to trigger the registers which will be storing the incoming flits. The VC register will be triggered at falling clock edge if the incoming flit propagates the link during high clock phase and vice versa.

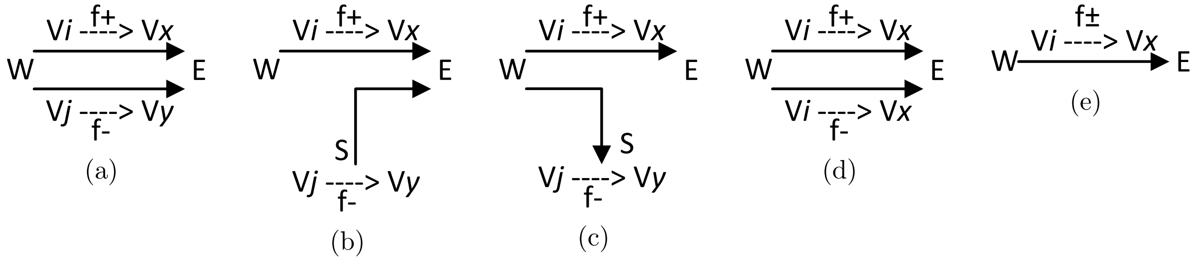


Figure 3: Alternative ways to propagate a flit through the DDRNoC router datapath.

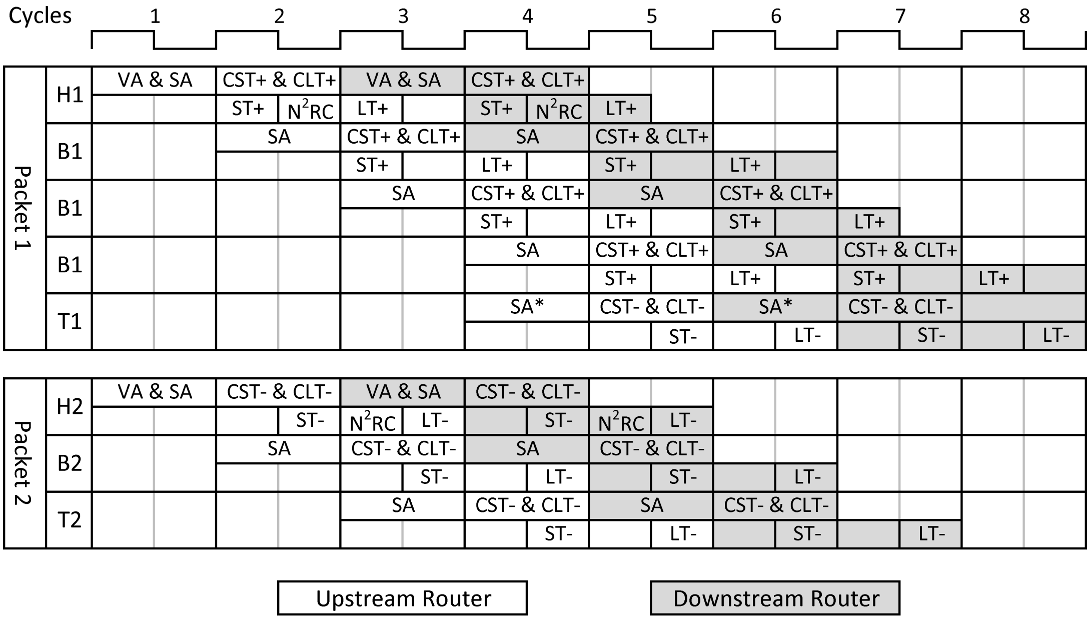
and their requested datapath is not allocated to any other packet during the same cycle. Figure 3, illustrates more precisely all alternative modes of the DDRNoC datapath usage. In Figure 3a, two flits in different VCs of the same input port are sent to the same output port in DDR (DDR_IN_en and DDR_LINK_en enabled). Figure 3b shows two flits from different input ports sent to the same output port in DDR (DDR_OUT_en and DDR_LINK_en enabled). Two flits from different VCs of the same input in DDR going to different outputs are illustrated in Figure 3c (DDR_IN_en is enabled). Figure 3d shows two flits of the same packet (same input VC) in DDR (DDR_VC_en and DDR_LINK_en enabled). Finally, a flit sent in SDR is depicted in Figure 3e.

In parallel to the ST of (up to) two flits, their control information ($FC+$ and $FC-$) is forwarded separately. As explained in the next paragraph, both $FC+$ and $FC-$ are switched during the first half of a cycle and traverse the link during the second half. This is possible as the minimum ST and LT delay in a DDRNoC router is half a cycle.

3.2 Timing

Despite operating at DDR or SDR, a single flit always stays in a datapath stage (ST or LT) for an entire cycle. In DDR mode, a flit utilizes either the high or the low phase of the clock. We call a flit using the high clock phase $flit^{+}$ and one using the low phase $flit^{-}$. A $flit^{+}$ is switched during the first half of a cycle and registered by the negative edge triggered output register $Reg-$, it uses again the high phase of the next cycle for LT and is stored at the input VC of the downstream router in the next negative clock edge. Similarly, a $flit^{-}$ uses for both ST and LT two consecutive low clock phases, it is registered in the positive edge triggered output register $Reg+$ and is stored in a VC buffer of the downstream router in the next positive clock edge. Finally, a flit switched at SDR ($flit^{\pm}$) uses a complete cycle for ST, is registered in $Reg+$, then performs LT for one cycle and is stored in a VC buffer of the downstream router at the next positive edge.

Besides data, a flit usually carries additional control bits that indicate flit type, allocated VC, or the computed next route. In a baseline SDR router, these control bits pass through ST and LT together with the data of a flit. On the contrary, in a DDRNoC router, the control bits of flit(s) travel before the data so as to initiate the SA in the downstream router a cycle earlier and reduce packet latency. During ST of flits $flit^{+}$, $flit^{-}$, their control bits $FC+$ and $FC-$ perform both ST and LT (half a cycle each). In effect, during the cycle $flit^{+}$ and $flit^{-}$ perform LT, their control bits are already in the downstream router and are considered in the SA and VA, saving one cycle. We call the above technique *control-forwarding.* Control-forwarding is possible in our router architecture for two reasons: firstly, due to the fact that switch and link traversal have a latency of half a cycle, and secondly, because a flit spends an entire cycle in each stage, although it could spend half a cycle. A flit does not enter a new datapath stage in the middle of a cycle to avoid misalignments with the SA in the downstream router, which anyway starts at the positive clock edge and takes an entire cycle to complete. Although the data of a flit are routed slower than they potentially could, sending the control of the flits faster,



*T1 is considered for SA with last B1.

Figure 4: Timing of two packets traversing two hops through the same input and output ports (different VCs) of two routers.

enables the downstream router to start SA a cycle earlier recovering the wasted time.

In order to perform the NRC in parallel with the VA and SA in a router with control forwarding, the entire destination address of a packet should be forwarded together with the rest of the control bits (earlier than the actual header flit). That would be quite costly as it would require for a 16×16 2D-mesh, 16 more link wires for the two flits in flight. To avoid this, we perform the NRC in the output of the upstream router, effectively, performing a routing computation for two hops ahead (Next-Next Route Computation: N^{2}RC). A flit takes half a cycle to traverse a link, during either the high or low clock phase, but it is sitting idle in an output register ($Reg+$, $Reg-$) for the other half of the cycle. N^{2}RC is performed during this idle half cycle. The N^{2}RC module is placed after the output registers and before the link multiplexer. When a head flit is registered in the output, N^{2}RC is performed for half a cycle, the result of the N^{2}RC overwrites the (two) respective header bits and is sent to the link multiplexer before link traversal. Note that the two output registers are triggered in different clock edges, so the same N^{2}RC module can be used by one of them in each half of the cycle.

For the header flit, SA speculates on the VA allocation result. Thereby, the VA can be performed in parallel with the SA. The header flit zero-load latency is then reduced by one cycle per hop when the speculation is successful.

Figure 4 illustrates the timing of two DDRNoC routers, putting all the above together. In the example, a packet of five flits and a packet of three flits traverse the same two routers. In the first router, VA and SA is performed during the first cycle for the head flits of the two packets. Subsequently, ST is performed for the header flits (ST$^+$ and ST$^-$), each using half of a cycle. In parallel, during cycle 2, control information for the head flits is traversing the switch (CST$^+$ and CST$^-$ for flit$^+$ and flit$^-$, respectively) and the link (CLT$^+$ and CLT$^-$). H1 traverses the switch in the high phase of clock 2 and is stored in the output register $Reg-$ at the falling edge of cycle 2 enabling N^{2}RC to be performed in the second half of cycle 2. Similarly, H2 traverses the switch in the low phase of clock 2, it is stored in the output register $Reg+$ at the rising edge of cycle 3 enabling N^{2}RC to be performed in the first half of cycle 3. LT is then performed by H1 and H2 after their N^{2}RC during the first and second half of cycle 3, respectively. In parallel during the same cycle (cycle 3), SA and VA are performed in the downstream router for the two header flits, as their control information has arrived a cycle earlier. The subsequent flits of the two packets follow, sharing the datapath of the two routers, having one cycle latency per ST and LT stage and throughput of two flits per cycle. Note, that packet 2 is two flits shorter than packet 1 and therefore the last two flits of packet 1 are routed in DDR mode through the two routers, similar to the example of Figure 3d. The SA in cycle 4 checks that the input port and the output port of packet 1 does not have any contention (because competing packet 2 has already propagated the switch), VC containing packet 1 has at least two flits stored (B1 and T1) belonging to the same packet and its output VC has at least two available credits. In cycle 5 as a result of previous SA, the DDR_VC_en signal to the input port where packet 1 is stored is set, along with a grant signal to the input VC. This indicates to the input VC control unit that in the current cycle two flits of the granted VC have been allowed access to the switch. As T1 is the last flit in the VC and it has already been granted access to the switch, there is no need to send any further arbitration requests to the SA. Further details of the SA are provided in section 3.4.2.

3.3 Zero load latency analysis

The zero load latency (ZLL) of a packet in the DDRNoC[3] is equal to one cycle for the first SA plus two cycles per hop for the first two flits (which include the head flit) and an additional half a cycle for each of the remaining flits[4]:

$$ZLL_{DDRNoC} = 1 + 2 * hops + (N-2)/2 \ cycles \tag{1}$$

where, *hops* is the number of hops traversed by the packet, and N the number of flits per packet.

In comparison, each flit in a typical 3-stage baseline SDR router passes through: SA/VA, ST, and LT having a throughput of one flit per cycle per port. The zero load latency for the baseline network would then be:

$$ZLL_{base} = 3 * hops + N - 1 \ cycles \tag{2}$$

A ShortPath router with pipeline bypass would require at best one cycle for routing and one for link traversal. The zero load latency for a ShortPath network would then be:

$$ZLL_{shortpath} = 2 * hops + N - 1 \ cycles \tag{3}$$

As mentioned in the introduction and confirmed in our evaluation, the ST/LT stages in a router can be clocked about 45% faster than a SDR baseline router and about 25% faster than ShortPath. The ST or LT delay is then about 70% and 80% the delay of the baseline SDR and ShortPath NoCs, respectively. Considering that the DDRNoC can have a clock period of 2× the ST/LT delay, then one DDRNoC cycle is equal to about 1.4 baseline cycles and 1.6 ShortPath cycles. In that case, zero load packet latency for the DDRNoC is similar to the baseline SDR and about 50% longer than ShortPath, as observed in Section 6.

3.4 DDRNoC Control

3.4.1 Virtual Channel Allocation

Despite the increasing load of the VA in a DDRNoC router, we choose to use the SDR baseline VA unmodified. That is a VA with round-robin priority-based output-first separable allocator as described by Becker and Dally [29]. For a router with P ports and V VCs per port, the VA uses $PV : 1$ and $V : 1$ arbiters for the output and input arbitration, respectively, and takes one

[3] We consider that the VC buffer size is sufficient to support the credit round-trip time, as explained in 3.4.3.

[4] In zero load, a single packet can send its flits in DDR having serialization latency of half a cycle per flit (Figure 3d). In case of contention, the serialization latency would increase at least to one cycle per flit.

cycle to complete allocation. In DDR mode, up to two head flits per cycle may arrive at an input port of a router, which (in the worst case) would result in ten new VA requests per cycle, compared to five in the baseline. Although this may lower the VA matching quality, we did not observe any significant performance drawbacks in our experiments for packet sizes based on application driven workloads.

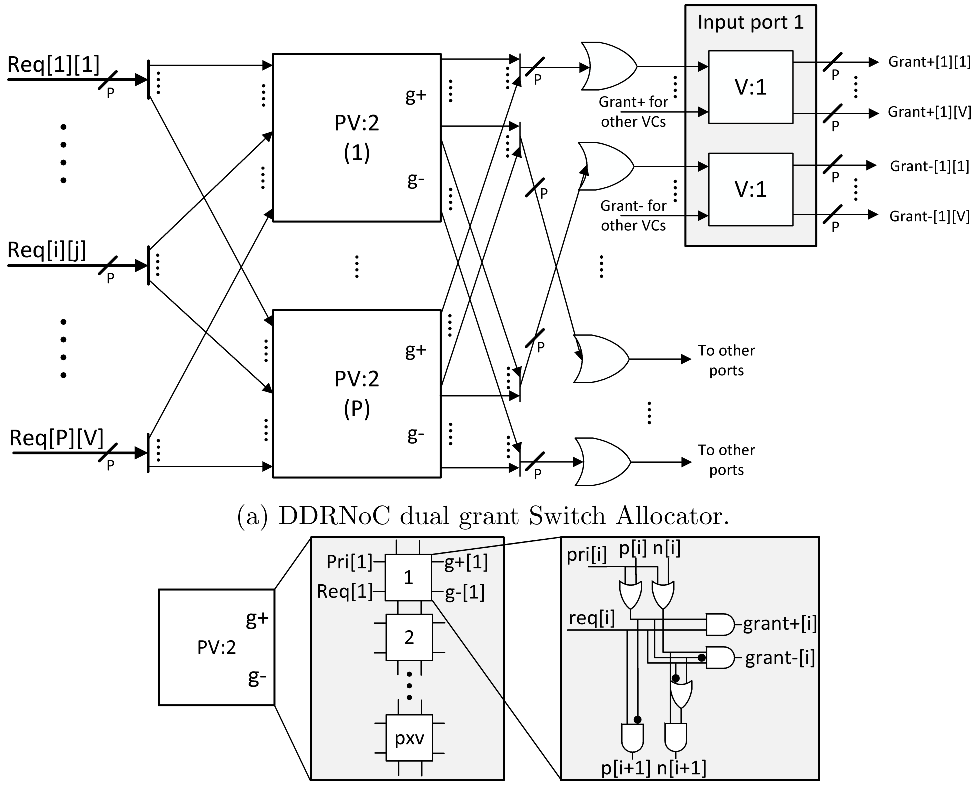
3.4.2 Switch Allocation

The DDRNoC routers, as well as the baseline SDR ones, use a single-cycle speculative SA to resolve contention among flits requesting the same input and output ports of the crossbar. The speculative SA gives higher priority to requests which have already been allocated a downstream VC. The DDRNoC SA, shown in Figure 5a is a modified version of the output-first separable allocator described in [29]. It makes two input and output arbitration decisions every cycle, one for the high ($g^+[i]$) and one for the low ($g^-[i]$) phase of the clock. The output arbiter for a single output port, shown in Figure 5b, accepts $P\times$V arbitration requests and a one-hot priority vector (which updates in round-robin) to generate two grant signals for each input request. At most one of these two grant signals would be asserted per input request. Moreover, at most one g^+ and one g^- signal would be asserted as a result of each output port arbitration. The second stage of the SA performs (input) arbitration among the VCs of an input port, which after the output arbitration might have two or more of its VCs granted access to the switch on the same half of the cycle. Two V:1 input arbiters per port are required, one to arbitrate among grants received for the high half of the cycle and the other among the grants received for the low half. After input arbitration a maximum of two grants (one per clock edge, $grant+$ and $grant-$) are asserted for each input port. The DDRNoC SA described so far is not yet able to allow two flits of the same packet to be switched in DDR mode. This is performed using additional logic appended to the above SA. In particular, based on the above grant decisions, a single packet, with an allocated downstream VC, will be allowed to send two flits through the switch in a cycle when all following conditions are met: (i) that packet has received a SA grant, (ii) more than one flits of the packet are available in the input VC, (iii) there is enough space in the downstream VC buffer, (iv) no other packet has been granted the same input or output of the switch. After SA, the Grant+ and Grant- signals are registered and the DDR enable signals are generated. These are the signals that control the DDR mode in each VC buffer (DDR_VC_en), input port multiplexer (DDR_IN_en), output port multiplexer (DDR_OUT_en), and output (DDR_LINK_en).

3.4.3 Flow control and minimum buffer size

The DDRNoC uses credit based flow control. Since flits from two different input VCs can be granted access to the switch in a single cycle, credits for up to two different VCs need to be transmitted to the upstream router simultaneously. This is implemented using one wire per VC for sending credits[5]. Moreover, a single VC can also forward two flits of a packet in a single cycle. In order to send back two credits to a single VC in a cycle, one additional wire per port is used to inform the upstream router that the credit counter of the VC receiving credit(s) should be incremented by 1 or 2. A credit consumed after switch grant, can successfully be received back in a minimum of 4 cycles after SA of the flit in the downstream router. This means that using 4 registers per VC would be sufficient to cover the credit-round-trip-time (t_{crt}). This is true however only if a single VC sends one flit per cycle. In case a single VC sends two flits at DDR (Fig. 3d) then it occupies 2 downstream buffers per cycle increasing the the minimum VC buffer size requirement to 8 (although the t_{crt} remains the same). In comparison, the 3-stage baseline NoC has a t_{crt} of 5 cycles and uses VC buffers of 5 flits.

[5] This is sufficient for 4 VCs per port considered in our implementation, but as the number of VCs per port increases, it becomes more scalable to send the ids of the two granted VCs, instead of a bitmask.



(a) DDRNoC dual grant Switch Allocator.

(b) PV:2 output arbiter.

Figure 5: Block diagram of the DDRNoC Switch Allocator.

3.5 Discussion

We discuss below some of the DDRNoC design decisions and other alternatives. As opposed to a baseline SDR router, the critical path of DDRNoC router is moved to the datapath (2× the ST or LT delay) allowing the DDRNoC control to have significant slack (about 30% of the cycle). In our current design, this slack in the DDRNoC control is largely unexploited. Although the DDRNoC SA is more complex than the baseline SA, it is almost as slow as the VA block, which is the same for both the baseline and the DDRNoC. Consequently, more advanced SA and VA modules can be explored in the DDRNoC without affecting its cycle time. Another observation is that instead of operating at DDR using both clock edges, similar throughput could be achieved by a NoC that has a clock twice as fast as the DDRNoC clock and makes allocation decisions every two cycles using a pipelined SA. Such design would have many similarities with the proposed DDRNoC although it might be slightly less challenging to implement. However, offering two cycles for the allocation decisions, even if the allocators were pipelined, would affect the efficiency of the speculative SA. Moreover, the clock distribution power would be double than that of the DDRNoC and hence the total NoC power would be about 20% higher. Even compared to the baseline SDR NoC, the total power consumption of such a network would be expected to be about 9% higher due to a 45% faster clock. Moreover, packets in such a router would spend 4 cycles per hop because of increased router pipeline depth, which would require 6 registers per VC buffer to cover the t_{crt} of 6 cycles.

4 Implementation and experimental setup

In our experiments, networks are implemented in Register Transfer Level (RTL) abstraction to accurately measure operating frequency, area and power consumption and in addition modelled in SystemC at a cycle accurate level and with modified GARNET to obtain network performance results for longer simulations and for full system results, respectively.

In our evaluation Section 5, DDRNoC performance, power and area costs are measured in comparison to a typical 3-stage baseline SDR architecture [21] using both RTL implementations and SystemC models. Such a baseline is straight-forward to implement in RTL to obtain detailed power and area results. In Section 6, comparison with other current state-of-the-art related NoCs is performed only in terms of performance (throughput, latency) using cycle-accurate SystemC models. The implementation details of the DDRNoC router as well as of the other networks used for comparison are listed in Table 1.

Both the DDRNoC and the baseline SDR network are implemented in CMOS FDSOI (Fully Depleted Silicon on Insulator) 28 nm technology standard cell libraries of 1.10V and 0.95V. The designs were placed and routed (P&R) with Cadence Design Systems SoC Encounter. Link latency at 1.1V is 80 ps/mm, using repeaters[6]. We consider tiles with their longest side being 2.85 mm based on the Chip multiprocessors parameters used by Sewell et al. after scaling down to 28 nm (CPU core + 32 kB L1 instruction and data caches + a 512 kB L2 cache slice) [30]. Finally, registers in the datapath support clock gating to reduce power consumption when idle.

Power analysis is performed simulating post-P&R netlists of the NoCs in Questasim with back-annotated delays. The baseline SDR NoC is warmed up for 500 cycles and evaluated for 6000 cycles during power estimation. The DDRNoC is warmed up for 250 cycles and evaluated for 3000 cycles as it has double activity per cycle. Gate-level switching activity for each router in the NoC is recorded in a VCD file which was then used to get power estimates of the entire NoC using Synopsys Primetime PX.

Performance analysis is performed by injecting synthetic and application-driven traffic, as well as carrying out full system simulations. The following synthetic traffic patterns are considered: (i) uniform random (UR), (ii) hotspots (HS) with 25% of the traffic going to 4 hotspots, one at each NoC corner, and the rest of the traffic being uniform random, (iii) nearest neighbour (NN), and (iv) bit reverse (BR). Half of the injected packets are considered to be data packets composed of 80 bytes (5 flits) and the other half control packets of 16 bytes (1 flit). In addition, traces based on application driven workloads are obtained using SynFull [31]. These traces capture the application behaviour of various PARSEC [32] and SPLASH-2 [33] benchmarks including messages generated by the cache coherence protocol and message dependencies. Simulations run until completion, all below 100 million cycles, generating packets 16 or 80 bytes for a 32 node (4×8) network. In these experiments, average packet latency is measured per benchmark. Power analysis for such long simulations is not possible in practice. So, we estimated energy per transferred bit using synthetic traffic that exhibits similar characteristics with the applications traffic and then used it to calculate EDP. Finally, we evaluate the impact the increased DDRNoC network throughput has to the execution time of applications performing full system simulations using GEM5 and GARNET [34, 35]. In these experiments we use the same above benchmarks but simulate smaller 16 node (4×4) systems. Moreover, for all the experiments, we consider a Network Interface (NI) which can support the same data-rate as the network being evaluated, ensuring that the NI is not the limiting factor.

[6] The same technology is considered, in the comparison Section 6, to estimate the maximum operating frequency of related works.

[7] This is sufficient for Baseline SDR, ShortPath and RapidLink NoC because they have a credit-round-trip-time of 5 cycles. It is also sufficient for a DDRNoC that handles up to 5 flits long packets.

[8] This SA was used instead of the output-first separable allocator with P:1 and V:1 output and input arbiters of [29] because (i) it has better performance, (ii) it is not more complex than the VA used so it doesn't affect cycle time, and (iii) it is similar to the DDRNoC SA.

[9] A PV:2 arbiter has about 50% higher delay than a PV:1 used by the baseline SA because each block has a delay of three, rather than two 2-input gates. That makes DDRNoC SA slower than baseline SA, but, without affecting router's frequency as DDRNoC control has a slack.

Table 1: Implementation parameters of the DDRNoC, the baseline SDR NoC, ShortPath [11], RapidLink [17] and RapidLink with pipelined Links (RapidLink2) [18].

Design	Baseline SDR (1.1V & 0.95V)	DDRNoC (1.1V & 0.95V)	ShortPath NoC [11]	RapidLink [17]	RapidLink2 [18]
Router architecture	3-stage router: VA/SA/NRC, ST, LT	2-stage router: ST, LT (VA/SA parallel to LT of upstream router)	4-stage router: VA, SA1, SA2/ST, LT. Dynamic pipeline-stage bypassing	2 ShortPath routers sharing the same links in DDR	2 ShortPath routers sharing the same links in DDR
Flow ctrl	Credit based flow control				
Link	137 bits: 128b data, 3b flit type, 2b VC-id, 4b credits	147 bits: 128b data, 2×3b flit type, 2×2b VC-id, 2×2b N^{2}RC, 4b+1b credits	138 bits: 128b data, 3b flit type, 2b VC-id, 4b flit-credits, 1b pkt-credit	138 bits: 128b data, 3b flit type, 2b VC-id, 4b flit-credits, 1b pkt-credit. With DDR support	2 pipeline stages using DDR DS-EB with VC support [18] 142 bits: 128b data, 3b flit type, 2b VC-id, 4b flit-credits, 1b pkt-credit, 4b DS-EB flow control. With DDR support
Latency-per-hop (cycles)	Min: 3 Max: 4	Min: 2 Max: 4	Min: 2 Max: 4	Min: 1.5 Max: 3.5	Min: 2.5 Max: 4.5
VC Configuration	4 VCs per input port.			2 VCs per input port per subnetwork	2 / 4 VCs per input port per subnetwork
Buffer size	5-flit flip-flop based VC buffers. [7]				8 flit buffers per VC
VC allocator	Output-first separable allocator, Round-Robin priority, PV:1 and V:1 arbiters for output and input arbitration, respectively [29]		V:1 input arbitration and P:1 output arbitration. VC Allocation Request Queue depth = 10		
Switch allocator	Speculative output-first separable allocator, Round-Robin priority, PV:1 and V:1 arbiters for output and input arbitration, respectively[8]	Speculative output-first separable allocator, Round-Robin priority, PV:2 and V:1 arbiters for output and input arbitration, respectively[9]	2-stage pipelined SA. Input arbitration SA1 V:1. Output arbitration SA2 P:1. SA Request Queue depth = 8. Merged SA2 and ST		
Routing	XY routing with NRC	XY routing with N^{2}RC	XY routing with NRC		

Table 2: Post place & route area and operating frequencies of the DDRNoC and the baseline SDR NoC.

Voltage	Design	Area (No. of Gates)	Max Op. Frequency (GHz)
1.10 V	Baseline	250568	2.02
	DDRNoC	260646	1.47
0.95 V	Baseline	209117	1.67
	DDRNoC	209835	1.11

5 Evaluation Results

This section presents the implementation results of the DDRNoC and the baseline SDR NoC, an evaluation of their performance and energy efficiency using synthetic traffic and traces of application-driven workloads, as well as overall application execution time running running on systems that employ the networks.

5.1 Implementation Results

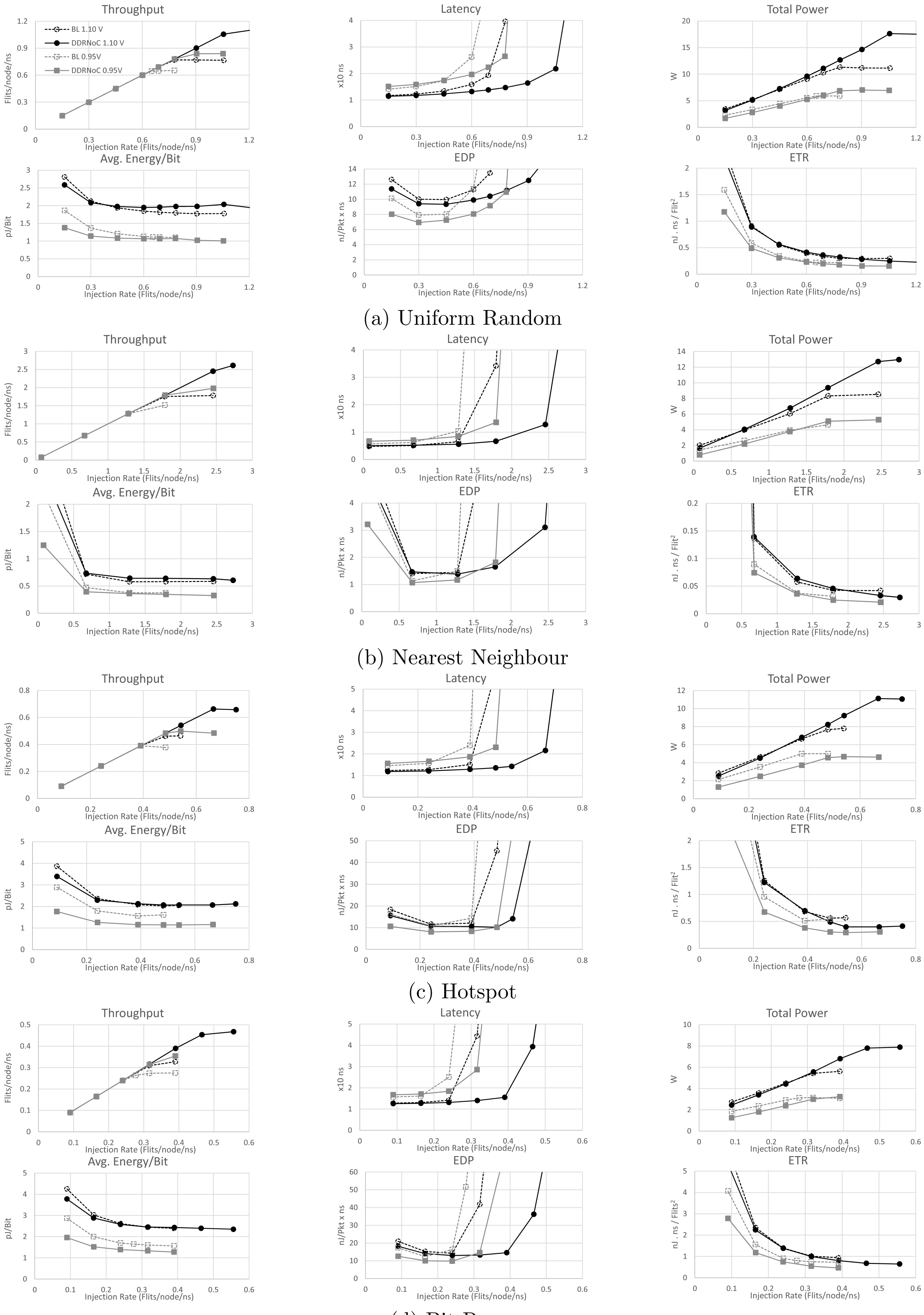
Table 2 summarizes the post P&R results of a single DDRNoC and baseline router (tile), taking into account link delays. These tiles are then used as building blocks of the evaluated NoCs. DDRNoC requires up to 4% more gates mostly due to its slightly wider datapath (crossbar and links), an additional register-multiplexer pair per output port and more complex SA. The input port, VC buffers and their control logic have negligible area overhead (less than 1%) while the VA is exactly the same as in the BL router. Its operating frequency is 73% of its iso-voltage baseline in 1.1V and 55% in 0.95V. As confirmed by our evaluation, this yields for the 1.1V DDRNoC larger improvements in performance rather than in energy efficiency, compared to its baseline. On the other hand, 0.95V DDRNoC exhibits better energy efficiency and slightly lower performance gains versus its iso-voltage baseline.

Analyzing the delays of individual modules in the DDRNoC and baseline SDR (1.1V) routers, we observe the following. The critical path of the baseline router is in the VA and is $495ps$. Switch and link have a delay of $340ps$ in both the baseline and the DDRNoC. Then, DDRNoC has a cycle time of $680ps$, its VA is the same as in baseline ($495ps$) and its SA has a higher delay of $550ps$ due to the use of PV:2 output arbiters. The above confirm our previous statements. DDRNoC throughput can be up to 45% higher than the baseline $(\frac{1\ flit/340ps}{1\ flit/495ps} = 1.45)$. The ST/LT delay is about 70% of the baseline control $(\frac{340ps}{495ps} = 0.69)$. The cycle time of DDRNoC is about 40% $(\frac{680ps}{495ps} = 1.38)$ higher than the baseline. Finally, there is still significant slack in the DDRNoC control to improve allocation ($680ps - 550ps = 130ps$ slack).

5.2 Evaluation using Synthetic Traffic

Using synthetic traffic we evaluate 8×8 2D-mesh DDRNoC and baseline networks composed of the above implemented routers at 1.1V and 0.95V and present our results in Figure 6. Performance is evaluated in terms of network throughput and packet latency. We further report total power consumption and energy efficiency measured in energy per transferred bit, energy delay product (EDP) and energy throughput ratio (ETR).

Compared to the SDR baseline at the same voltage, DDRNoC at 1.1V improves throughput by 45%. At 0.95V that improvement is 30% due to the larger gap between the baseline and DDRNoC operating frequencies. Similarly, DDRNoC 1.1V packet latency is better even in low injection rates (2-4%) with an exception in nearest neighbor traffic where its 5% higher because of low average hop count of this traffic. DDRNoC at 0.95V has 7% (UR, HS) to 20% (NN, BR) higher latency than the 0.95V baseline, due to a slower clock. In both cases, DDRNoC has lower packet latency at higher injection rates due to lower contention.



(a) Uniform Random

(b) Nearest Neighbour

(c) Hotspot

(d) Bit Reverse

Figure 6: DDRNoC evaluation compared to the baseline SDR NoC in terms of throughput, end-to-end packet latency, NoC power consumption, energy per transferred bit, Energy Delay Product (EDP), and Energy Throughput Ratio (ETR).

Total power consumption is in general proportional to the traffic served. At low injection rates, the total power consumption of a DDRNoC network is better compared to its iso-voltage baseline, 10-20% at 1.1V and 25-45% at 0.95V. That is due to the fact that in these injection rates the idle power[10] is a significant fraction of the total power. At the baseline saturation point, power consumption is up to 10% higher for 1.1V DDRNoC, but lower at 0.95V. Finally, at maximum DDRNoC throughput and at 1.1V the 45% higher throughput costs 40-55% more power. At 0.95V, 30% better DDRNoC throughput costs only 3-18% more power, while in HS traffic it is even 6.5% lower than the baseline power. That is because in HS traffic injection rates are quite low and hence idle power is a large contributor to the total power.

Results of energy per transferred bit are quite mixed for 1.1V, where DDRNoC is better (up to 17%) at low injection rates and worse (up to 10%) at higher injection rates. For the 0.95V implementations, energy per transferred bit is always better for DDRNoC up to 26% to 43% in all traffic patterns.

Compared to 1.1V baseline, 1.1V DDRNoC has always lower EDP (8% to 22%) except at low injection rates of NN traffic where the baseline is 4% lower. At 0.95V, DDRNoC is always more energy efficient than its baseline, having an EDP that is 6% to 25% lower at low injection rates and 20% to 40% lower before the baseline saturation.

DDRNoC ETR is also better than its baseline with few exceptions at low injection rates of 1.1V designs. Especially at the DDRNoC saturation point, for 1.1V ETR reduces by 26% to 31% and for 0.95V by 27% to 45%.

An interesting comparison is between the 1.1V baseline and the 0.95V DDRNoC. This DDRNoC design point attempts to capitalize the slack between control and datapath gaining mostly energy efficiency rather than performance. Still throughput is 8% to 10% better for all traffic patterns, while packet latency is 26% to 40% higher due to the slower clock. On the other hand, power consumption is 40% to 60% lower. This translates to 40% to 60% lower energy per transferred bit, 20% to 45% lower EDP, and 37% to 60% lower ETR.

5.3 Evaluation using application-driven traffic

Using SynFull, we measure packet latency and EDP of a 32-node (4×8) DDRNoC and baseline SDR NoCs for various PARSEC and SPLASH-2 benchmarks. In order to stress networks' throughput, in these experiments we considered for all networks 32-bit datapaths, rather than 128-bits and SynFull packet generator step of 400ps. Average throughput does not provide any useful insight and therefore is not reported since for all networks the traffic generated by a benchmark is entirely delivered to its destination. Network performance in terms of throughput is reflected in packet latency, too.

Figures 7 and 8 report the average packet latency and EDP per benchmark for each of the two networks implemented in 1.1V and 0.95V, as well as the geometric mean of all benchmarks. DDRNoC at 1.1V reduces packet latency by 2% to 25% and EDP by 5% to 24% compared to the SDR baseline operating at the same voltage. In 0.95V, DDRNoC has slightly lower performance but better energy efficiency compared to its iso-voltage baseline; packet latency increases only up to 8% and in some cases decreases by 7%, but EDP is 9% to 21% lower. Finally, the low voltage DDRNoC compared to the 1.1V baseline has up to 30% increase in latency, but EDP is always 32% to 45% better.

Overall across all benchmarks, compared to their iso-voltage baselines, 1.1V DDRNoC reduces packet latency by 13% and EDP by 14%, while 0.95V DDRNoC has similar latency and 15% lower EDP. A 0.95V DDRNoC compared to a 1.1V baseline has 14% higher packet latency and 38% lower EDP.

[10] Idle power is the power consumption of a NoC when it is not serving any packets. DDRNoC has lower idle power due to its slower clock and lower clock distribution power.

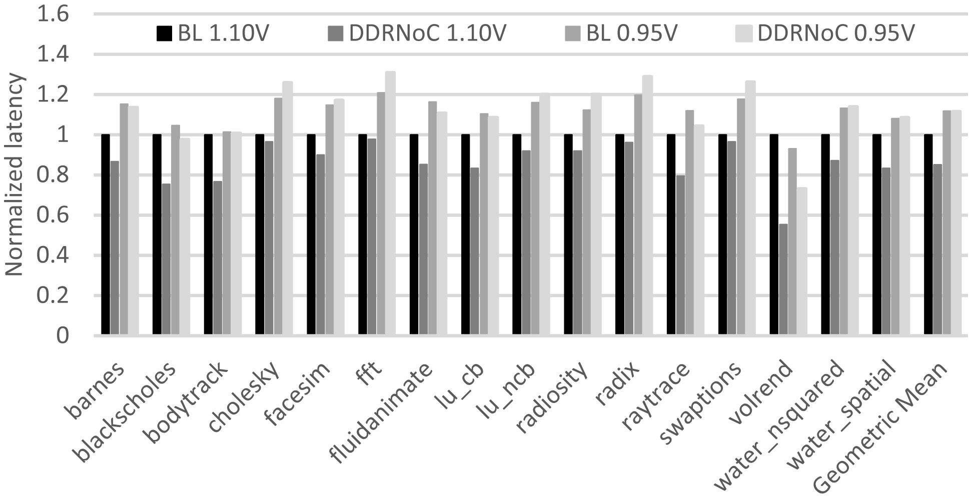


Figure 7: Normalized packet latency of DDRNoC and baseline SDR NoC for PARSEC and SPLASH-2 benchmarks.

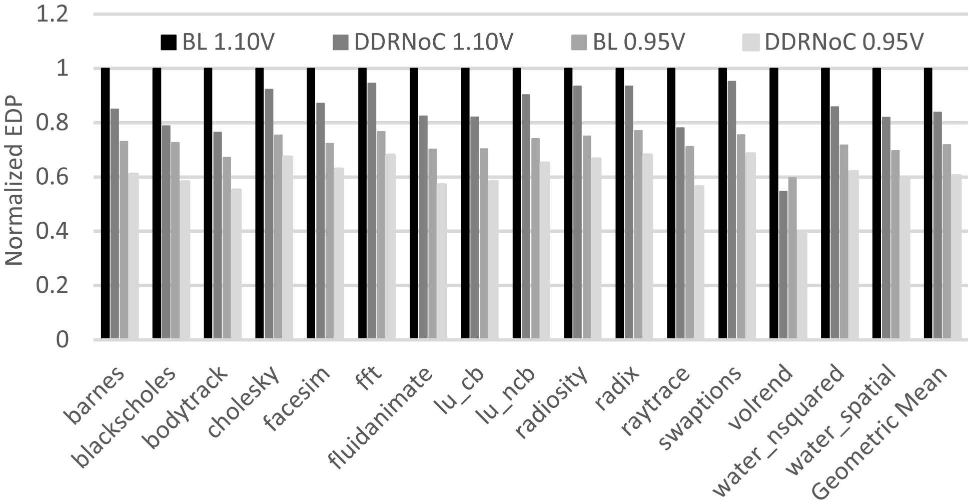


Figure 8: Normalized EDP of DDRNoC and baseline SDR NoC for PARSEC and SPLASH-2 benchmarks.

5.4 System-level evaluation

In order to appreciate the impact of the improved DDRNoC network throughput to the application execution time and energy efficiency we finally performed full system simulations using GEM5 [34] system simulator, Ruby based memory subsystem models, and BL and DDNoC 1.1V models based on modified GARNET [35]. The detailed system configuration is presented in Table 3. We run medium-sized multithreaded benchmarks from PARSEC-2.1 benchmark suite and measure the execution time of the parallel phase of the application called the Region of Interest (ROI)[11]. In order to estimate system power and EDP we used McPAT considering 28nm technology.

Figure 9 shows the normalized execution time with respect to the BL network of the ROI of 10 different PARSEC2.1 benchmarks. The largest network we could simulate was a 4×4 2D-mesh. Despite the small network size we can observe that DDRNoC reduces application execution time by up to 11% and on average by 6% in the benchmarks used. As discussed in the introduction, the benefits of improving network throughput have been studied in more

[11] We were not able include *ferret*, *rtview* and *x264* benchmarks because bugs in the GEM5 simulator would cause the simulations to crash.

Table 3: System and NoC configurations for Gem5 full system simulations.

System Parameters	
Cores	16 in-order Alpha ISA, 2.0GHz
Private L1 I/D cache	32 KB, 4 way set associative
Shared distributed L2 cache	8 MB, 8 way set associative, 16 directories
Cache line size	64 Bytes
Replacement policy	Pseudo-LRU
Cache coherency	MESI protocol
Topology	4x4 2D mesh
Memory Size	1 GB
Network Parameters	
VNs, VCs	3 VNs, 2 VCs/VN 1 buffer per ctrl VC 4 buffers per data VC
Packet size	Ctrl: 2 flits Data: 18 flits
Links	32 bits, 1 cycle latency

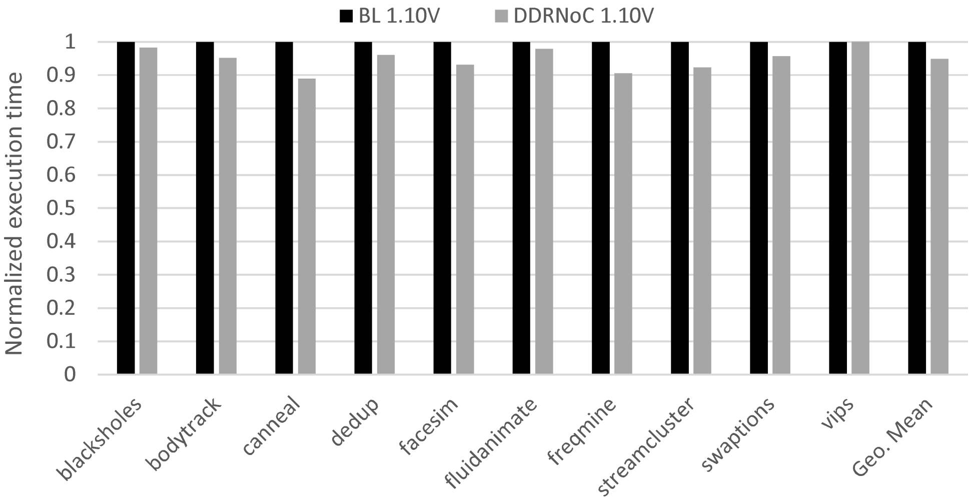


Figure 9: Normalized application execution time for PARSEC-2.1 benchmarks using full system simulations.

detail by Bakhoda et al. showing that doubling the network bandwidth improved execution time of applications up to 2× [26]. In addition, DDRNoC achieves lower EDP compared to the BL network by up to 20% and on average 10%, as shown in Figure 10.

6 Comparison

In this section, we compare the performance of DDRNoC against the current state-of-the-art SDR and DDR networks. More precisely, DDRNoC is compared in terms of latency and throughput against ShortPath [11], the fastest reported SDR NoC architecture, as well as against RapidLink [17] and its improved version denoted as RapidLink2 [18], which are the most recent NoCs that use DDR in parts of their datapath (the links). The SDR baseline used in the previous section is also shown here for reference. Network packet latency and throughput are measured using the same synthetic and application-driven traffic as in the previous section.

ShortPath is reported to be the fastest SDR NoC router [11], faster than SCORPIO [10]. At low injection rates it is able to bypass stages and route flits in a single cycle (plus a cycle for LT). At high injection rates however flits may need to go through all three router stages, requiring up to four cycles per hop. We modeled ShortPath behavior in cycle-accurate SystemC and further estimated its maximum operating frequency in 28nm technology as follows. According

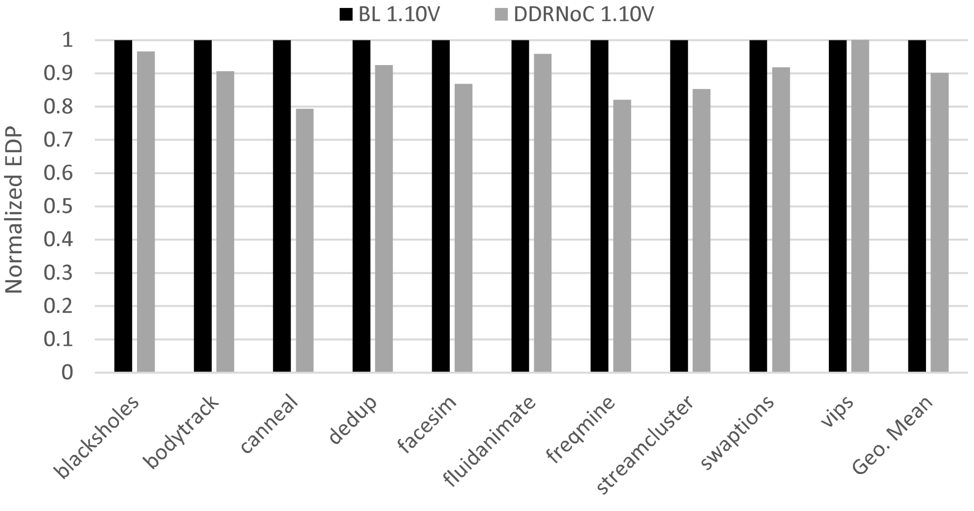


Figure 10: Normalized processor EDP for PARSEC-2.1 benchmarks using full system simulations.

to Psarras et al. the critical path of ShortPath is determined by the delays of credit-check (considering credits being already in a local register), an N:1 arbiter (for the second part of SA), two 2-to-1 multiplexers and the crossbar delay [11]. We partly implemented ShortPath in RTL to analyze the above path in 28 nm technology. In doing so, we estimated the critical path of ShortPath when implemented in the same technology used for the DDRNoC to be 420ps operating at 2.38GHz. Considering this frequency and the results of the SystemC model we measured ShortPath performance.

As discussed in Section 2, RapidLink allows two subnetworks to share links, which operate at DDR [17]. The assumption is that each link is fast enough to carry two flits in a single cycle without compromising the clock frequency of the router. Then, for some traffic patterns RapidLink can improve the throughput of the NoC it is applied to [17]. However, applying the RapidLink technique to an SDR architecture such as ShortPath would limit LT delay to only 210ps. Considering registers setup and propagation delays, the delay of a multiplexer for selecting each subnetwork, and wire delay of 80ps/mm that link cannot be longer than 1.25 mm[12]. Such link length is barely able to interconnect small tiles of cores that have only L1 caches. According to Turley et al. and after scaling to 28nm, even tiles of Cortex-A15 with L1 would need 1.5mm links for their interconnection [36]. In order for RapidLink to support 2.85mm links and tile sizes that fit a core with 32kB L1 instruction and data caches and 512kB L2 cache, like the DDRNoC, it needs to increase its cycle time to 2×340 ps matching the DDRNoC operating frequency. The performance of RapidLink applied to a ShortPath architecture is analyzed considering link lengths of 2.85 mm as it is the case for all other networks in our comparison.

RapidLink2 addresses the above drawback of the initial design by modifying further the NoC datapath (the LT) [18]. Besides splitting the router in two subrouters, RapidLink2 pipelines the network links. Thereby, it is able to operate at the maximum operating frequency allowed by the router it is applied to. Our measurements indicate that a RapidLink2 design applied to ShortPath routers requires links to be pipelined with two Dual-Stream Elastic Buffers with VC support [18] (3 stages) in order to achieve the minimum cycle time (420ps). According to our experiments, this increases the link energy per transferred bit (for 128 bits wide flit) by 2.5×[13]. We compare RapidLink2 with a DDRNoC design that exhibits similar router datapath

[12] For 0.95V circuits the results are similar, as the Shortpath router has a cycle time of 570ps and wire delay is 110ps/mm. Considering setup and propagation logic delay a Shortpath router with RapidLink would be able to support up to 1.2mm links

[13] Splitting LT in two stages would require 1.8× more link energy per transferred bit.

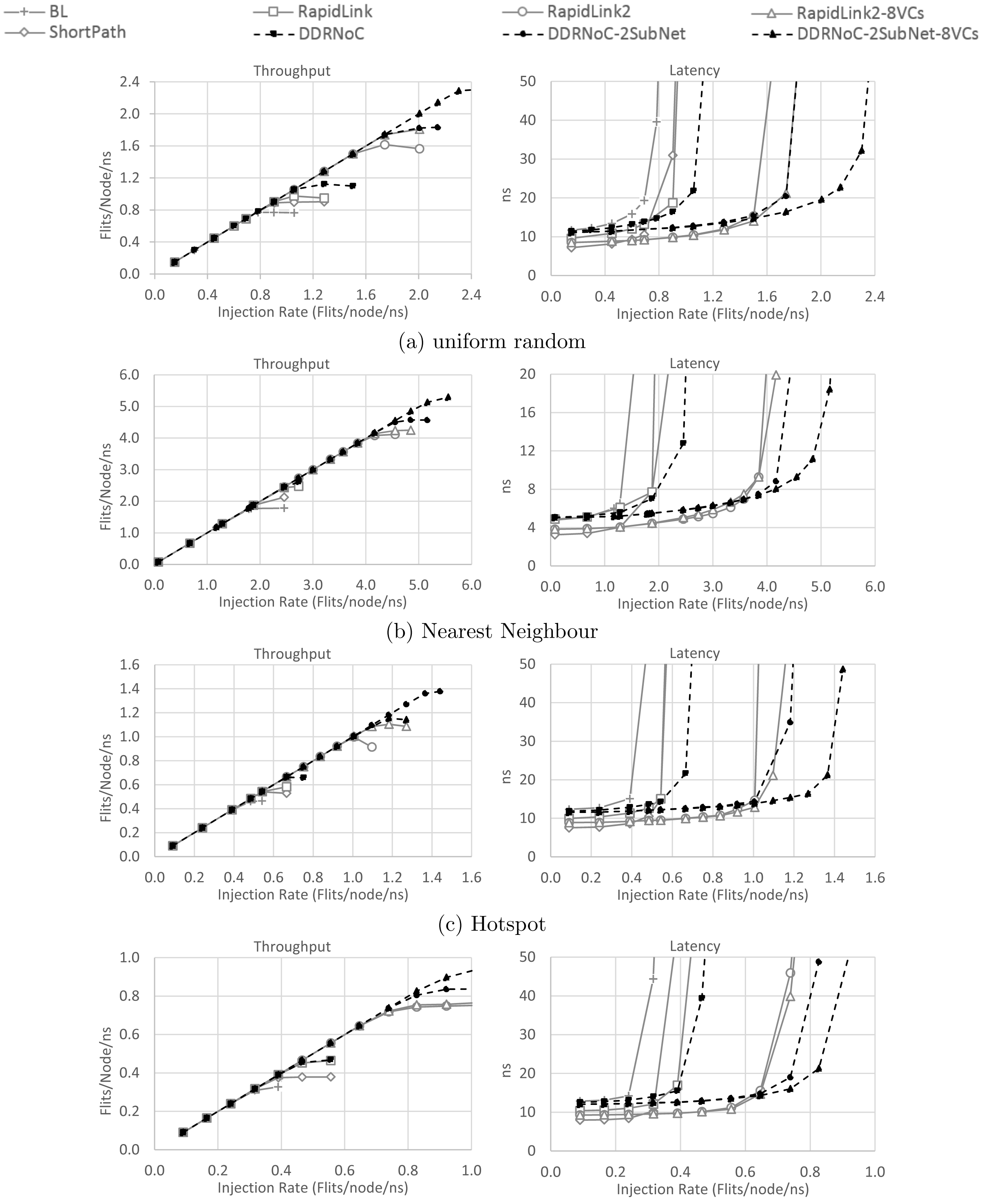
modifications. We consider a DDRNoC with two subnetworks (denoted as DDRNoC-2SubNet); its routers are split to two subrouters similar to RapidLink, while the links are replicated (rather than pipelined) avoiding the Rapidlink2 energy overheads to the links. In the 28nm technology used for our implementation, replicated links can be supported in the low resistance intermediate Bx metal layers using a small fraction of the available wires – 5% to 10% depending on spacing and shielding.

Figure 11 shows the average packet latency and throughput of DDRNoC in comparison to the SDR baseline, ShortPath, RapidLink and RapidLink2 (applied to ShortPath) 8×8 2D mesh networks injecting the same synthetic traffic as in the previous section. For a fair comparison with RapidLink and RapidLink2, we also present results for a DDRNoC with two subnetworks.

In uniform random traffic, ShortPath has about 18% better throughput than the SDR baseline. Using RapidLink on ShortPath further improves throughput by another 8% but gives away most of the latency gain because its frequency drops to 1.47 GHz to allow 2.85mm long links. DDRNoC has about 15% better throughput than RapidLink and 25% better than Shortpath. RapidLink2 improves RapidLink throughput by 65%. However, DDRNoC with 2 subnetworks is 15% and 27% better compared to RapidLink2 when using 4 and 8 VCs, respectively. Here, the larger the number of VCs the larger the throughput improvement for the DDRNoC over RapidLink2; that is because more VCs allow better sharing of the same datapath (in DDR) between packets. In nearest neighbor traffic, DDRNoC is able to improve throughput by 20% and 6% over ShortPath and RapidLink, respectively. Moreover DDRNoC with two subnetworks is 10% and 25% better than RapidLink2 with 4 and 8 VCs, respectively. In hotspot traffic, RapidLink improves Shortpath throughput by 5%. While DDRNoC is about 25% better than ShortPath. RapidLink2 doubles Shortpath throughput, but DDRNoC with two subnetworks is 15% and 23% better than RapidLink2 for 4 and 8 VCs, respectively. Finally, in bit-reverse traffic DDRNoC achieves 7%, and 21% better throughput than RapidLink and ShortPath, respectively. RapidLink2 improves throughput of RapidLink by 61%. DDRNoC with two subnetworks has 13% and 23% better throughput compared to RapidLink2 for 4 and 8 VCs, respectively.

For all synthetic traffic patterns, packet latency trends are similar. ShortPath has the lowest packet latency and RapidLink2 and RapidLink come next. ShortPath, RapidLink2 and RapidLink have at best, at low injection rates, 65%, 75% and 85% of the DDRNoC latency, respectively. The only exception is in the nearest neighbor traffic where RapidLink is not able to exploit the half-cycle LT due to the small hop count, and then at low injection rates its latency is similar to the DDRNoC.

We further compared the average packet latency of the eight designs in 8×4 2D mesh topologies using application-driven traffic extracted from PARSEC and SPLASH-2 benchmarks via SynFull. In order to stress designs with multiple subnetworks, in these experiments we considered for all networks 32-bit datapaths, rather than 128-bits and SynFull packet generator step of 200ps (400ps for *fft* and *radix* benchmarks which cause network saturation). Figure 12 depicts the average packet latency results of the networks. Baseline and ShortPath saturate in the *barnes*, *bodytrack* and *fft* benchmarks. Baseline further saturates in *radix*, *volrend* and *water_spacial* benchmarks. RapidLink saturates with *fft* and *radix* benchmarks. DDRNoC does not saturate for any of the benchmarks and therefore is used to normalize the other results. RapidLink2 and DDRNoC-2SubNet networks do not saturate either. The geometric mean is calculated excluding the saturated results of the baseline, ShortPath and RapidLink networks and shows that for the remaining benchmarks ShortPath and RapidLink have 7% and 2% higher average packet latency compared to the DDRNoC. Average latency of DDRNoC-2SubNet is 2% and 4% higher than Rapidlink2, when using 4 and 8 VCs, respectively. This is because many of the benchmarks still inject low traffic loads which favor the RapidLink networks based on ShortPath, which uses pipeline bypassing. However, when considering benchmarks which demand more network throughout, for example for *barnes* and *bodytrack* benchmarks [31], DDRNoC-



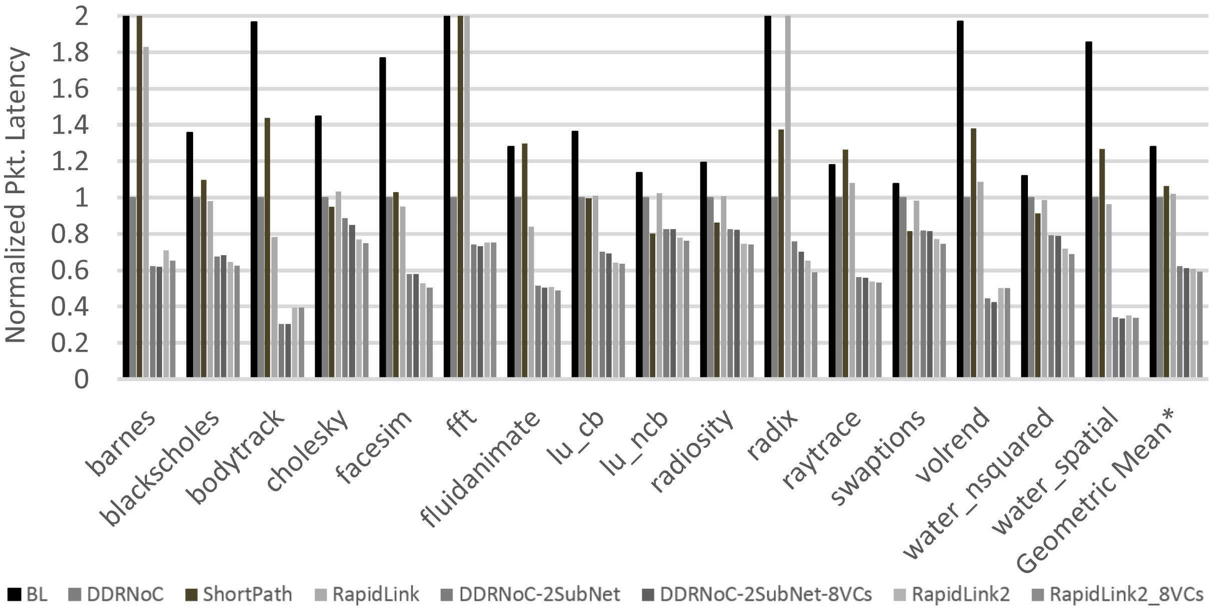
(a) uniform random

(b) Nearest Neighbour

(c) Hotspot

(d) Bit reverse

Figure 11: An 8x8 DDRNoC throughput and latency comparison with our baseline network, ShortPath network and a RapidLink network based on ShortPath router architecture. Moreover, DDRNoC-2SubNet is compared to RapidLink2 with 4 and 8 VCs.



*Geometric Mean does not include results for saturated networks (for benchmarks they exceed the vertical axis).

Figure 12: Comparison of average packet latency (normalized to the DDRNoC results) for PARSEC and SPLASH-2 benchmarks. SynFull packet generator step is set to 200 ps, except for *fft* and *radix* benchmarks which is set to 400 ps as they saturate because of higher traffic loads.

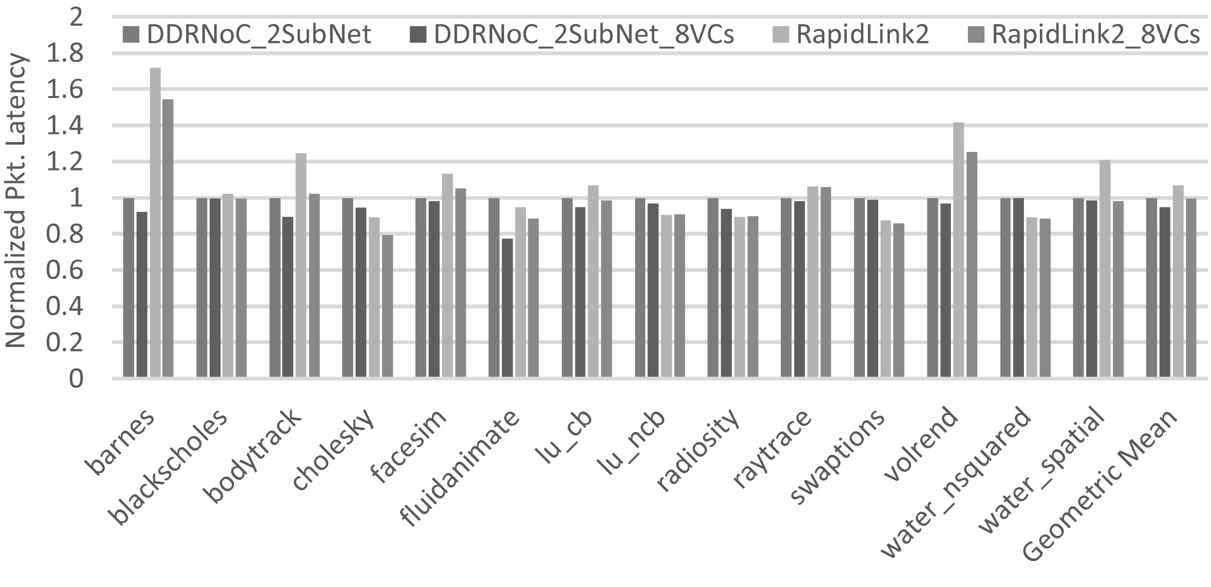


Figure 13: Comparison of average packet latency (normalized to the DDRNoC-2SubNet results) for PARSEC and SPLASH-2 benchmarks. SynFull packet generator step is set to 100 ps. *fft* and *radix* benchmarks are omitted as they cause congestion in all networks. Results are shown only for Rapidlink2 and DDRNoC with two subnetworks since these are the only ones that do not saturate.

2SubNet variants show 6% to 22% lower average packet latency compared to RapidLink2. In order to further stress the networks, we also evaluated the DDRNoC-2SubNet and the RapidLink2 8×8 2D mesh networks for the same benchmarks reducing the SynFull packet generator step to half (100 ps) as shown in Figure 13. *FFT* and *radix* benchmarks saturate for all networks. For the rest of the benchmarks, it can be observed that 4 and 8 VC variants of the RapidLink2 networks have on average 7% and 5% higher average packet latency than DDRNoC-2SubNet networks.

In general, RapidLink can only offer significant throughput improvements in systems with very small tile sizes. As links get longer its clock frequency drops giving away any performance gains. RapidLink2 pipelines the links using DS Elastic Buffers to operate the network at a frequency based on the critical path of the router instead of the link delay. DDRNoC has up to 20% better throughput than RapidLink while its 2-subnet variants have up to 27% higher throughput than RapidLink2. This is expected as DDRNoC allows all competing flits to pass at DDR through every part of a router datapath (within a subnetwork for DDRNoC-2SubNets), as opposed to RapidLink and RapidLink2 that use DDR to resolve contention only at the link and only between flits of different subnetworks. In our evaluation, RapidLink and RapidLink2 have 15% and 25% lower latency, respectively, compared to the DDRNoC for some synthetic traffic patterns because they use ShortPath routers. Moreover, DDRNoC has 20-25% better throughput than ShortPath because the rate at which it routes flits is determined purely by the datapath delay, while the critical path of ShortPath, despite the improved cycle time, is still partly determined by the control. ShortPath has about 35% lower latency than DDRNoC at low injection rates, due to pipeline bypassing. As discussed in the introduction, packet latency is more important for the performance of workloads that exhibit small transfers at low loads; for more demanding workloads that push the network close to its saturation point, throughput is more important. Such workloads are more representative of systems that run concurrent scale-out applications [19, 20]. Targeting such systems, DDRNoC offers throughput that is up to 45% higher than a baseline 3-stage SDR NoC, up to 25% better than the current state-of-the-art SDR NoC and up to 27% better than a NoC that uses DDR links.

7 Conclusion

DDRNoC is a new on-chip interconnection network that uses double-pumped routers. It is based on the observation that conventional SDR 2D mesh routers have significant slack on their datapath stages. DDRNoC uses this slack offering two flits to share the same datapath within a cycle at DDR. Thereby, DDRNoC supports up to 27-45% higher throughput compared to SDR NoCs and up to 27% better than a network with only DDR links. Control forwarding is employed to reduce DDRNoC packet latency, which compared to SDR routers with pipeline bypassing still suffers at low injection rates. Alternatively, a low-voltage DDRNoC implementation reduces power consumption by 40%, at the cost of 26-40% higher latency, still however offering similar or slightly higher throughput. Finally, DDRNoC reduces energy per transferred bit, EDP and ETR due to a slower clock and its low voltage implementation further improves energy efficiency offering a substantially better energy-performance trade-off.

References

[1] H. Esmaeilzadeh, E. Blem, R. St. Amant, K. Sankaralingam, and D. Burger, “Dark silicon and the end of multicore scaling,” in *ISCA*, 2011, pp. 365–376.

[2] S. Borkar, “How to stop interconnects from hindering the future of computing!” in *2013 Optical Interconnects Conf.*, May 2013, pp. 96–97.

[3] M. B. Taylor, J. Kim, J. Miller, D. Wentzlaff, F. Ghodrat, B. Greenwald, H. Hoffman, P. Johnson, J.-W. Lee, W. Lee *et al.*, "The raw microprocessor: A computational fabric for software circuits and general-purpose programs," *IEEE micro*, vol. 22, no. 2, pp. 25–35, 2002.

[4] Y. Hoskote, S. Vangal, A. Singh, N. Borkar, and S. Borkar, "A 5-ghz mesh interconnect for a teraflops processor," *IEEE Micro*, vol. 27, no. 5, pp. 51–61, 2007.

[5] S. Bell, B. Edwards, J. Amann, R. Conlin, K. Joyce, V. Leung, J. MacKay, M. Reif, L. Bao, J. Brown *et al.*, "Tile64-processor: A 64-core soc with mesh interconnect," in *IEEE Int. Solid-State Circuits Conf.*, 2008, pp. 88–598.

[6] P. Salihundam, S. Jain, T. Jacob, S. Kumar, V. Erraguntla, Y. Hoskote, S. Vangal, G. Ruhl, and N. Borkar, "A 2 tb/s 6 x 4 mesh network for a single-chip cloud computer with dvfs in 45 nm cmos," *IEEE J. of Solid-State Circuits*, vol. 46, no. 4, pp. 757–766, April 2011.

[7] B. Grot, J. Hestness, S. W. Keckler, and O. Mutlu, "Express cube topologies for on-chip interconnects," in *HPCA*, Feb 2009, pp. 163–174.

[8] C. H. O. Chen, N. Agarwal, T. Krishna, K. H. Koo, L. S. Peh, and K. C. Saraswat, "Physical vs. virtual express topologies with low-swing links for future many-core nocs," in *NOCS*, 2010, pp. 173–180.

[9] M. Hayenga and M. Lipasti, "The NoX router," in *MICRO-44*, 2011.

[10] B. K. Daya, C.-H. O. Chen, S. Subramanian, W.-C. Kwon, S. Park, T. Krishna, J. Holt, A. P. Chandrakasan, and L.-S. Peh, "Scorpio: A 36-core research chip demonstrating snoopy coherence on a scalable mesh noc with in-network ordering," in *Int. Symp. on Computer Architecuture*, ser. ISCA '14, 2014, pp. 25–36.

[11] A. Psarras, I. Seitanidis, C. Nicopoulos, and G. Dimitrakopoulos., "ShortPath: A Network-on-Chip Router with Fine-Grained Pipeline Bypassing," *IEEE Trans. on Computers*, vol. 65, no. 10, pp. 3136–3147, 2016.

[12] J. Kim, D. Park, T. Theocharides, N. Vijaykrishnan, and C. R. Das, "A low latency router supporting adaptivity for on-chip interconnects," in *Design Automation Conf. (DAC)*, 2005, pp. 559–564.

[13] S. Rao, S. Jeloka, R. Das, D. Blaauw, R. Dreslinski, and T. Mudge, "Vix: Virtual input crossbar for efficient switch allocation," in *Design Automation Conf. (DAC)*, 2014, pp. 103:1–103:6.

[14] J. Kim, J. Balfour, and W. Dally, "Flattened butterfly topology for on-chip networks," in *MICRO-40*, 2007, pp. 172–182.

[15] J. Balfour and W. J. Dally, "Design tradeoffs for tiled cmp on-chip networks," in *Int. Conf. on Supercomputing*, 2006, pp. 187–198.

[16] F. Gilabert, M. Gómez, S. Medardoni, and D. Bertozzi, "Improved utilization of noc channel bandwidth by switch replication for cost-effective multi-processor systems-on-chip," in *NOCS*, 2010.

[17] A. Psarras, S. Moisidis, C. Nicopoulos, and G. Dimitrakopoulos, "Rapidlink: A network-on-chip architecture with double-data-rate links," in *2016 IEEE International Conference on Electronics, Circuits and Systems, ICECS*, 2016, pp. 93–96.

[18] ——, “Networks-on-chip with double-data-rate links,” *IEEE Trans. on Circuits and Systems*, vol. 64, no. 12, pp. 3103–3114, 2017.

[19] P. Lotfi-Kamran, B. Grot, and B. Falsafi, “Noc-out: Microarchitecting a scale-out processor,” in *2012 45th Annual IEEE/ACM Int. Symp. on Microarchitecture*, Dec 2012, pp. 177–187.

[20] A. Psathakis, V. Papaefstathiou, N. Chrysos, F. Chaix, E. Vasilakis, D. Pnevmatikatos, and M. Katevenis, “A systematic evaluation of emerging mesh-like cmp nocs,” in *ANCS*, 2015, pp. 159–170.

[21] W. Dally and B. Towles, *Principles and Practices of Interconnection Networks.* Morgan Kaufmann Publishers, 2004.

[22] C. Nicopoulos, V. Narayanan, and C. R. Das, *Network-on-Chip Architectures - A Holistic Design Exploration*, ser. Lecture Notes in Elect. Eng. Springer, 2010, vol. 45.

[23] H. Kim, A. Vitkovskiy, P. V. Gratz, and V. Soteriou, “Use it or lose it: Wear-out and lifetime in future chip multiprocessors,” in *MICRO-46*, 2013, pp. 136–147.

[24] M. A. Anders, “High-performance energy-efficient noc fabrics: Evolution and future challenges,” in *IEEE/ACM NOCS*, Sept 2014.

[25] S. Vangal, A. Singh, J. Howard, S. Dighe, N. Borkar, and A. Alvandpour, “A 5.1ghz 0.34mm2 router for network-on-chip applications,” in *IEEE Symp. on VLSI Circuits*, June 2007, pp. 42–43.

[26] A. Bakhoda, J. Kim, and T. M. Aamodt, “Throughput-effective on-chip networks for manycore accelerators,” in *MICRO-43*, 2010, pp. 421–432.

[27] A. K. Mishra, R. Das, S. Eachempati, R. Iyer, N. Vijaykrishnan, and C. R. Das, “A case for dynamic frequency tuning in on-chip networks,” in *MICRO-42*, 2009, pp. 292–303.

[28] J. Xu, W. Wolf, and W. Zhang, “Double-data-rate, wave-pipelined interconnect for asynchronous nocs,” *IEEE Micro*, vol. 29, no. 3, pp. 20–30, May 2009.

[29] D. U. Becker and W. J. Dally, “Allocator implementations for network-on-chip routers,” in *Conf. on HPC Net., Stor. and Analysis*, ser. SC ’09, 2009, pp. 52:1–52:12.

[30] K. Sewell, R. G. Dreslinski, T. Manville, S. Satpathy, N. R. Pinckney, G. Blake, M. Cieslak, R. Das, T. F. Wenisch, D. Sylvester, D. Blaauw, and T. N. Mudge, “Swizzle-switch networks for many-core systems.” *IEEE J. Emerg. Sel. Topics Circuits Syst.*, vol. 2, no. 2, pp. 278–294, 2012.

[31] M. Badr and N. E. Jerger, “Synfull: Synthetic traffic models capturing cache coherent behaviour,” in *ACM/IEEE ISCA*, June 2014, pp. 109–120.

[32] C. Bienia, “Benchmarking modern multiprocessors,” Ph.D. dissertation, Princeton, NJ, USA, 2011.

[33] S. C. Woo, M. Ohara, E. Torrie, J. P. Singh, and A. Gupta, “The splash-2 programs: Characterization and methodological considerations,” *SIGARCH Comp. Archit. News*, vol. 23, no. 2, pp. 24–36, 1995.

[34] N. Binkert, B. Beckmann, G. Black, S. K. Reinhardt, A. Saidi, A. Basu, J. Hestness, D. R. Hower, T. Krishna, S. Sardashti, R. Sen, K. Sewell, M. Shoaib, N. Vaish, M. D. Hill, and D. A. Wood, “The gem5 simulator,” *SIGARCH Comput. Archit. News*, vol. 39, no. 2, pp. 1–7, Aug. 2011.

[35] N. Agarwal, T. Krishna, L. S. Peh, and N. K. Jha, “Garnet: A detailed on-chip network model inside a full-system simulator,” in *IEEE Int. Symp. on Performance Analysis of Systems and Software*, April 2009, pp. 33–42.

[36] J. Turley, “Cortex-a15 ”eagle” flies the coop,,” *Microprocessor Report*, vol. 24, no. 11, pp. 1–11, Nov 2010.