Performance Evaluation and Design Trade-offs for Wireless-enabled SMART NoC

Karthi Duraisamy and Partha Pratim Pande
School of School of Electrical Engineering & Computer Science
Washington State University, Pullman USA.
{kduraisa, pande}@eecs.wsu.edu

Abstract – **SMART (Single-Cycle Multi-hop Asynchronous Repeated Traversal) NoC architectures enable single cycle data transfers, even between the physically far apart nodes. However, enabling single cycle hops over long distance restricts the achievable clock frequency of the system. In other words, increasing the NoC clock frequency lowers the number of hops that can be traversed in a single-cycle in a conventional SMART NoC. In this work, we demonstrate that by integrating wireless links and a novel look-ahead request mechanism in the SMART NoC, it is possible to enable low-latency and energy efficient data transfers, even when the system is designed with high clock frequencies. For the various applications considered in this work, the wireless-enabled SMART (WiSMART) NoC achieves on an average 33% reduction in message latency compared to the wireline SMART NoC. This network level improvement translates into 16% savings in full system energy-delay-product.**

Keywords—Wireless NoC; Single-Cycle Multi-Hop NoC

I. Introduction

Data exchanges in traditional NoC architectures are multihop in nature and hence involve high latencies [1][2]. The high network latencies eventually lead to long waits for memory accesses, stalling processor cycles and result in execution time penalties. The SMART (Single-Cycle Multi-hop Asynchronous Repeated Traversal) NoC paradigm is an efficient solution to minimize the latency of the traditional NoC architectures [3][4]. By using a router-bypass control mechanism and wires with asynchronous (clockless) repeaters, SMART enables single cycle data transfers between the NoC tiles that are physically far apart (only the repeaters are clockless while the routers and cores are synchronous). However, integrating the SMART control mechanism can restrict the clock frequency of a NoC. More specifically, with increasing *HPCmax* values, the clock frequency of the NoC decreases linearly (*HPCmax* is the maximum number of hops that can be bypassed within one clock cycle) [3]. Moreover, the router overhead in a SMART NoC grows quadratically with increasing *HPCmax* [5]. Hence, in order to enable high clock frequency and low router overhead, it is necessary to design NoC architectures with low *HPCmax*. However, as demonstrated later, designing SMART NoCs with the conventional mesh topology and employing a low *HPCmax* limits the achievable performance and scalability.

On the other hand, Wireless NoC (WiNoC) is an emerging paradigm to design high bandwidth and energy efficient communication backbone for manycore chips [6]. A recent study on emerging on-chip interconnects concluded that the wireless and surface-wave interconnects are also more power and cost efficient than the on-chip optical links [7]. Moreover, when compared to surface-wave interconnects, the on-chip wireless technology is more mature and is completely CMOS compatible [6][7]. In this work, we demonstrate that by integrating the SMART mechanism with mm-wave wireless links, one can design a scalable NoC architecture that can operate with high clock frequencies.

II. Related Work

There are two principal ways to enhance the performance of the conventional NoCs: (i) lowering the inter-router hop counts and (ii) lowering the router-stage latency. To lower the inter-router hop counts, high radix NoC architectures have been proposed [10][11][12]. These high radix NoCs connect a large number of cores to a single router and employ multiple long-range links (uses topologies such as flattened butterfly). However, routers incorporating a large number of ports are complex to design, dissipate high power and require 4-5 pipeline stages [12][4]. Following the principles of small-world graphs, the hop counts of a mesh NoC is reduced by inserting a few application-specific long-range links [2]. Moreover, power-law based small-world networks enable design of robust high performance low-power NoCs [8][9]. A router microarchitecture enabling single cycle hops is discussed in [13]. Router bypass techniques to enable single cycle hops are discussed in several previous works [14][15].

Unlike the above-mentioned NoCs, a SMART NoC can establish single cycle connections even between the physically distant (multiple hops apart) routers, dynamically depending on the communication requirements [3][4]. To accomplish such flexible long-range connections, SMART NoCs use router-bypass setup networks along with wires incorporating clockless repeaters. In a SMART NoC, from each output port spans a set of dedicated SMART-hop setup request (SSR) links that are multi-drop in nature. The SSR network is of two types (1D and 2D) and is used to forward the bypass requests to the intermediate routers, one cycle before the actual data flit arrives [3]. Despite its advantages, the control mechanism used in the SMART NoC gives rise to high router connection overheads mainly owing to the flit width of the multi-drop SSR links [5]. In order to lower the connection overheads, a SMART NoC with reduced wires (we refer this NoC as RSMART NoC) is proposed in [5]. In a RSMART NoC, the control information is exchanged using a dedicated SSR-mesh network (lying in parallel to the data transfer mesh network) and a set of pre-SSR control wires. The flit widths of these pre-SSR wires are much smaller than that of the SSR wires in SMART and hence RSMART NoCs require lower connection overheads than the SMART NoCs.

In SMART/RSMART NoCs, using high *HPCmax* values can restrict the operating frequency to 1-2GHz [3]. In this work,

This work was supported, in part by the US National Science Foundation (NSF) grants CNS 1564014, CCF 1514269 and CCF 1162202.

978-3-9815370-8-6/17/$31.00 ©2017 IEEE

we demonstrate that by integrating WiNoC and a SMART control mechanism with low *HPCmax* values, it is possible to design high performance interconnect architectures. As shown later, the wireless-enabled SMART architectures are more scalable with increasing system sizes and clock frequencies when compared to the wireline SMART NoCs.

III. Wireless enabled RSMART NoC Architecture

In this section, first we outline the motivation for designing a wireless-enabled SMART NoC through the analysis of the data transfer times in the RSMART NoC with varying *HPCmax*. As explained in Fig. 1, there are four router pipeline stages in a RSMART NoC namely; local arbitration, pre-SSR transmission and arbitration, SSR transmission and single-cycle multihop link traversal [5] (at zero loads [16] the local arbitration can be avoided to save one cycle). Traversing all these four stages is called as traversing a SMART hop. Figs. 2(a) and (b) show the average SMART hops and the zero load latency [16] under uniform random traffic for 1D and 2D RSMART mesh NoCs with varying *LPCmax* values (*LPCmax*= *HPCmax*×Link length per hop). We have considered a standard 16×16mm die with four different system sizes (8×8, 16×16, 24×24 and 32×32). As shown, for all the considered system sizes, the message transfer latencies monotonically increase with increasing *LPCmax*. This trend is mainly attributed to the fact that for increasing *LPCmax* values the NoC clock frequency decreases linearly, starting from the clock frequency values for *LPCmax=4mm* (2.75GHz with 1D and 2.25GHz with 2D [3]). The reduction in SMART hops achieved by increasing *LPCmax* (as shown in Fig. 2(a)) does not compensate the increase in the network latency caused by the reduced clock frequency. Similar trends were observed with traffic arising from real applications (shown later in Section IV.B) and also with the original SMART NoC proposed in [3]. Hence, from this analysis we can conclude that it is more beneficial to design SMART NoC architectures with high clock frequencies and low *HPCmax* values than with low clock frequencies and high *HPCmax* values. Thus, the clock frequency of the NoC should only be restricted by the allowed power budget and the *HPCmax* must be determined using the given NoC clock frequency (instead of determining NoC clock frequency restricted by the opted *HPCmax* value). The systems shown in [12][17][18] are examples for NoCs designed with high clock frequencies.

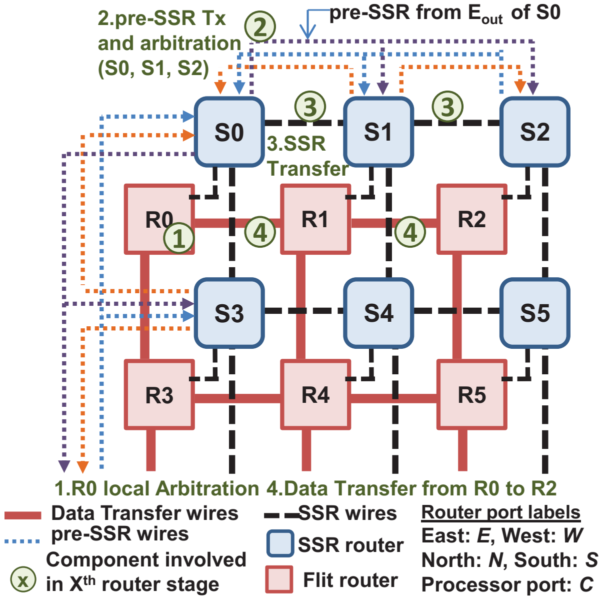
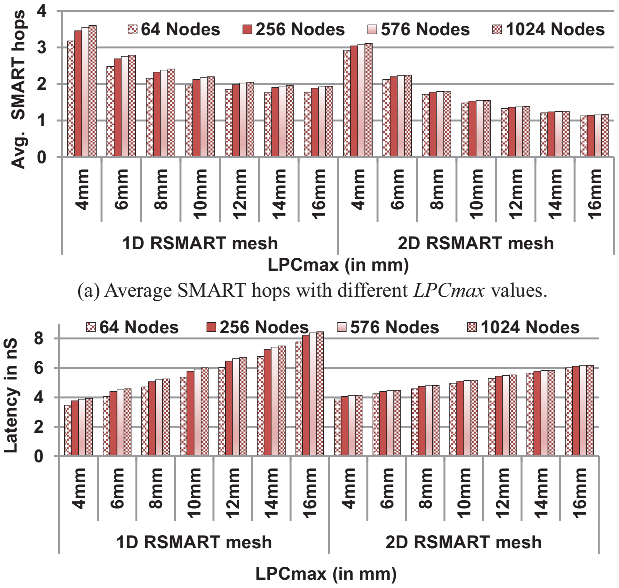

2.pre-SSR Tx and arbitration (S0, S1, S2)
2
pre-SSR from E_{out} of S0
S0
S1
S2
3
3
3.SSR Transfer
R0
1
4
R1
4
R2
S3
S4
S5
R3
R4
R5
1.R0 local Arbitration 4.Data Transfer from R0 to R2
Data Transfer wires
pre-SSR wires
SSR wires
SSR router
Flit router
x
Component involved in X^th router stage
Router port labels
East: *E*, West: *W*
North: *N*, South: *S*
Processor port: *C*
Stage 4 bypass connectivity: at R1: W_{in} →E_{out}, at R2: W_{in} →C_{out}

Fig. 1. 1D RSMART mesh labeled with the router stages involved in transmitting data from R0 to R2. The pre-SSRs are only shown for the nodes in first row and first column. *HPCmax* value used is 3. In stage 1, the message wins the local arbitration at R0. At stage 2, a single-bit pre-cursor request is forwarded from S0 to S1 and S2, along the pre-SSR wires. In the same stage, the pre-cursor request competes with other received pre-SSRs at S1 and S2. Upon successful pre-SSR arbitrations, a detailed bypass request is forwarded during stage 3 from S0 to S1 and S2 along the SSR wires. Using received SSR, routers R1 and R2 setup the desired bypass connectivity at stage 4, enabling the message to travel straight for port C_{in} of R0 to C_{out} of R2.

However, as shown in Fig. 2(a), the data-transfers in SMART/RSMART NoCs designed with low *HPCmax* values involve high number of SMART hops when a conventional mesh topology is employed. For such NoCs, we posit that it is beneficial to further enhance the architecture by incorporating additional on-chip wireless communication shortcuts (single-cycle long–range communication links). For this purpose, we consider on-chip wireless interfaces transmitting data over a 20mm range within one cycle [8] and operating in one of the five non-overlapping frequency bands centered on 30, 60, 90, 140 and 200 GHz. Next, we explain the design of this proposed NoC, called as WiSMART NoC.

A. *Placement of Wireless Interfaces in WiSMART NoC*

In a WiSMART NoC, few of the flit routers are provided with a wireless interface (WI). We follow a region based WI placement strategy to minimize both the number of WIs and the average SMART hop count. Minimizing the number WIs reduces the area and power overheads and also enables fast channel access times with low Medium Access Control (MAC) overheads [19]. In this region based WI placement strategy, we divide the NoC into multiple non-overlapping squares and place a set of WIs in each square (at least one WI for each wireless



(a) Average SMART hops with different *LPCmax* values.

(b) Zero load latency under uniform random traffic.

Fig. 2. Performance of RSMART mesh NoCs with varying *LPCmax*

channel) so that the average hop count is reduced. Since each square requires at least one WI per channel, the number of non-overlapping squares used is limited by the number of WIs allocated to a wireless channel. This in turn depends on the adopted MAC protocol [19]. Fig. 3 shows the placement of WIs in a 64-core WiSMART NoC employing 5 wireless channels and four squares of size 4×4.

By employing squares of side *HPCmax* and placing omnidirectional WIs in the center of each square, communication between any two nodes can be accomplished within three hops (2 SMART wireline hops and a wireless hop). For larger systems, one can further reduce the number of WIs by forming super-squares (16-tile super-squares in Fig. 3). In this strategy, sets of four squares are grouped together to form super-squares and for each super-square a maximum of two WIs per channel is used. However, employing super-squares increases the average hop counts. For e.g. in a 576 core NoC made of 4 super-squares and employing *HPCmax*=6 (thus 16 squares of side 6), a maximum of 4 SMART hops is required to communicate between any two nodes.

A. *Data Transfer in WiSMART NoC*

As shown in Fig. 2(b), the network latencies of the 1D and 2D RSMART NoCs are close to each other for small *HPCmax* values. However, the 1D RSMART NoCs require much lower connection overheads than the 2D RSMART [5]. Hence we primarily employ a 1D RSMART control network for WiSMART (with 2-bit pre-SSR wires that are explained later). The short range WiSMART data-transfers in which the communicating nodes present in the same or adjacent squares, are handled only through the wireline links. For the long-range communications, the message is first forwarded from the source node to a source WI using wireline links. The source WI then broadcasts the message using the wireless channel. Finally, a destination square (or super-square) WI receives the message and forwards it to the final destination node using wireline links. The wireline transfers in the WiSMART are similar to that of the XY routed 1D RSMART data transfers [5], except for the look-ahead request which is explained later.

Field	bit width
No., of Hops	$[log_2(1+N)]$
Is header Flit	1
Eject flag	1
Source ID	$[log_2(N)]$

Table. II. WiSMART SSR fields.

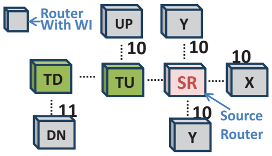


Fig. 4. pre-SSR values for WIs located in the different positions.

B. *Selection of Source WI*

In the WiSMART NoC, whenever a source WI becomes unavailable (due to failures or because the wireless channel is busy), the data flits are re-routed from the source WI to the final destination along wireline XY paths. Depending on the selection of the source WI, this re-routing mechanism can cause deadlocks in long-range data transfers. For example, if a message transmitted from node 11 to 45 in Fig. 3, selecting node 10 which is the physically closest WI to node 11 as the source WI can lead to a deadlock. In case of a WI failure at node 10, the message will simply move back and forth between nodes 10 and 11. With this in mind, given a set of source and destination co-ordinates, the rules that must be met by the source WI co-ordinates are given in Table. I. Thus, source WI is the nearest WI to the source node, satisfying the conditions in Table. I. Following these rules, to communicate from 11 to 45, the source WI used is at 13.

C. *Transmission of pre-SSRs*

In WiSMART, for each message injected, first a 2-bit pre-SSR code is identified. The data transfers that are used to access the source WIs follow a pre-SSR code of either 10 or 11, depending on the position of the source WI (as shown in Fig. 4). All other wireline data transfers use a pre-SSR code of 01 while 00 indicates a no-request. If the source WI is located in the same row or in the same column as that of the source node, the pre-SSR code used is 10 (indicated by WIs *X* and *Y* in Fig. 4). For out ports with no nearest WIs in the same row, an XY turn is required to reach the source WI (viz., E_{out} of *SR* in Fig. 4). In such cases, to avoid the conflicts, data transfers that require an upturn (*E*→*N* or *W*→*N*) are indicated through code 10 (accessing WI at *UP* from *SR* in Fig. 4) while those require down

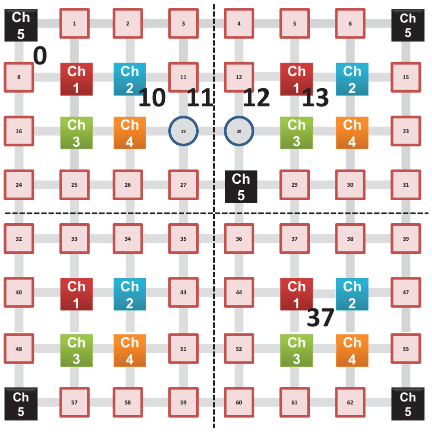


Fig. 3. A64-core WiSMART NoC employing four squares of size 4×4. Routers incorporating WI are labeled with their channel IDs. Following [19], the number of WIs in a channel is kept within 6 to lower the MAC latency. This NoC can also be seen as a 64-core WiSMART with 4 super-squares where each super-square is made of 2×2 squares.

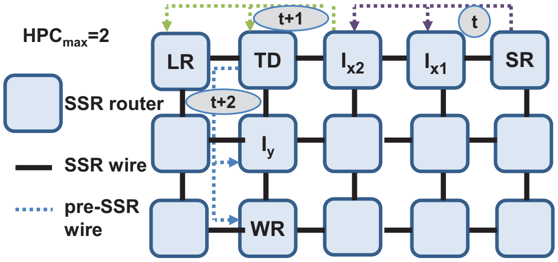


Fig. 5. Progression of pre-SSR from node *SR* to *WR* in 3 cycles using the look-ahead request mechanism. Only the necessary pre-SSR wires are shown.

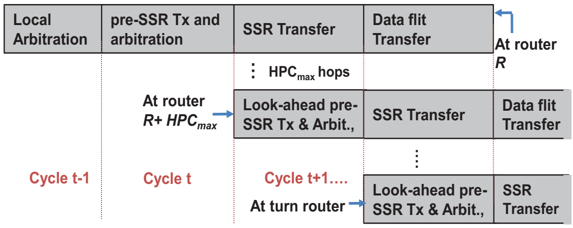


Fig. 6. Look-ahead request based routing in WiSMART NoC.

(*Sx*, *Sy*):Source (*Dx*, *Dy*):Destination, (*Wx*, *Wy*): Source-WI	
Given data	**Rules for source WI coordinates**
Sx < Dx	***Wx < Dx* and *Wx≥Sx***
Sx =Dx	***Wx=Sx***
Sx>Dx	***Wx >Dx* and *Wx≤Sx***
Sy < Dy	***Wy < Dy* and *Wy≥Sy***
Sy =Dy	***Wy=Sy***
Sy>Dy	***Wy >Dy* and *Wy≤Sy***

Table. I: The rules that must be satisfied by the source WI coordinates to avoid possible deadlocks arising from the re-routing mechanism used for handling WI failures.

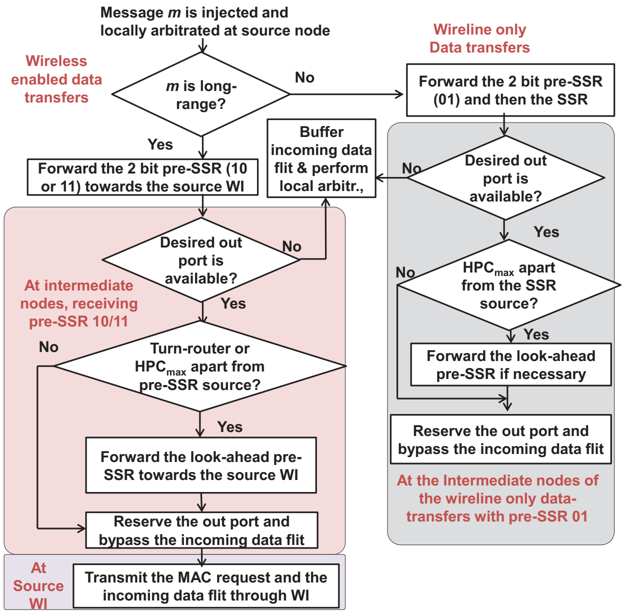

Message *m* is injected and locally arbitrated at source node
Wireless enabled data transfers
m is long-range?
No
Yes
Forward the 2 bit pre-SSR (10 or 11) towards the source WI
Buffer incoming data flit & perform local arbitr.,
Desired out port is available?
No
Yes
At intermediate nodes, receiving pre-SSR 10/11
No
Turn-router or HPC$_{max}$ apart from pre-SSR source?
Yes
Forward the look-ahead pre-SSR towards the source WI
Reserve the out port and bypass the incoming data flit
At Source WI
Transmit the MAC request and the incoming data flit through WI
Wireline only Data transfers
Forward the 2 bit pre-SSR (01) and then the SSR
No
Desired out port is available?
Yes
No
HPC$_{max}$ apart from the SSR source?
Yes
Forward the look-ahead pre-SSR if necessary
Reserve the out port and bypass the incoming data flit
At the Intermediate nodes of the wireline only data-transfers with pre-SSR 01

Fig. 7 Flowchart depicting the transsmission of WiSMART pre-SSR.

turns (*E*→*S* or *W*→*S*) are indicated through the pre-SSR code 11 (accessing WI router *DN* from *SR* in Fig. 4).

It is important to note that in the WiSMART NoC, along a particular direction, the source router (*SR* in Fig. 5) and all intermediate nodes (nodes between *SR* and turn router *TD* in Fig. 5) follow the same source WI. For example, in Fig. 5, receiving a value of 11 from nodes *SR*, I_{x1}, I_{x2} along their west pre-SSR wires would indicate a request for accessing the same source WI, node *WR*. Thus, by using the 2-bit pre-SSR data (if it is either 10 or 11) and ID of the node from which the pre-SSR originates[1], each intermediate node can identify the source WI to which the upcoming data flit is going to be forwarded. Exploiting this information and by using a set of status codes for each output port (3-bits codes representing bypass connections form the five different input ports), we employ a look-ahead request mechanism. We further clarify this mechanism through the example shown in Fig. 5. In this figure, at cycle t, a pre-SSR value of 11 is transmitted from the source node SR to the intermediate nodes that are within the *HPCmax* hops (nodes I_{x1} and I_{x2}). Upon receiving a pre-SSR of value 11 from the east direction, nodes I_{x1} and I_{x2} first identify that the request is to access the WI at router *WR*. Then, I_{x1} and I_{x2} set the status code of their W_{out} port, representing "bypass E_{in}→W_{out}". With this status code, the *Wout* port of routers I_{x1} and I_{x2} are now reserved till the end of data flit transfers, avoiding any local arbitration. In addition, I_{x2} also forwards a look-ahead pre-SSR to nodes *TD* and *LR*, at cycle *t+1*. Similarly, after receiving the pre-SSR, router *TD* sets the status of their S_{out} port as "bypass E_{in}→S_{out}" and forwards the look-ahead pre-SSR (of value 10) to nodes *Iy* and *WR*, at cycle *t+2*. Finally, *WR* receives the pre-SSR and broadcasts a wireless MAC request at cycle *t+3*, following the distributed MAC protocol [19]. The overall stages in the look-ahead request based routing used to access a WI scheme is shown in Fig. 6.

It should be noted that the look-ahead request is only applicable when the desired router ports[2] are not already reserved and the SSRs are not prematurely stopped[3]. If a desired output port is not being available, the look-ahead pre-SSR is avoided and the incoming data flit is buffered and forwarded to local arbitration. In case of prematurely stopped SSRs, reserved output ports are released in the next cycle.

Look-ahead pre-SSRs can also be extended to the wireline data transfers that are not used to access WIs (i.e., data transfers with a pre-SSR of 01). In such cases, since dimension-ordered XY routing is followed, we need to know the number of hops required in a dimension before forwarding look-ahead pre-SSR. This hop count information is only carried by the SSR wires and hence, the look-ahead pre-SSR needs to be delayed till the arrival of the SSR data. As shown in Table. II, we use $\lceil log_2(1+N) \rceil$ SSR wires to indicate the number of hops and $log_2(N)$ wires to indicate the position of the node, where the SMART hop originates. Hence, upon receiving SSR, each node can easily identify the number of remaining hops in a dimension and can forward a pre-SSR request. Fig.7 illustrates the WiSMART pre-SSR transmission.

D. Handling pre-SSR contention

To handle the contentions arising from different pre-SSRs requesting the same output port, following priority is used:

Reservation>Wireless pre-SSRs (10/11)>Wireline-only pre-SSR (01)

Within the wireless and wireline-only pre-SSRs, we use "*prio=local*" mechanism [3] to handle contentions by which the requests from the physically closer nodes are preferred over the relatively far nodes. Finally, if pre-SSRs coming from multiple directions request the same WI port in a router, we use the following priority rule to access the WI port:

$$\boldsymbol{Reservation > C_{in} > E_{in} > W_{in} > N_{in} > S_{in}}$$

Next, we incorporate the discussed routing in the WiSMART architecture and undertake a detailed performance evaluation.

IV. Performance Evaluation of WiSMART

In this section we evaluate the performance of the WiSMART NoC with respect to the RSMART mesh NoC. We also present a comparative performance evaluation with respect to a low hop-count high radix NoC topology i.e., flattened butterfly [12]. For this evaluation, we use a set of ten applications that exhibit a wide range of NoC traffic characteristics. More specifically, we use Canneal (CNL) and Fluidanimate (FLD) applications from PARSEC suite [20], FFT, Radix (RAD), LU and Water (WTR) from SPALSH-2 suite [21], Grappolo (GP) [22] and K-Means (KM), PCA and Linear Regression (LR) form Phoenix MapReduce suite [23]. Grappolo is used to identify the naturally existing communities in a real-world graph (community detection).

We use GEM5 full system simulator to obtain detailed processor and network information [24]. The memory system is comprised of private 64KB L1 instruction and data caches, one shared 16MB L2 cache (256KB distributed L2 per core) and employs the MESI directory protocol. The width of all data wires is same as the considered flit width of 128 bits.

[1] Source ID of is identified from the set of wires that carried the pre-SSR.

[2] Desired output ports are present in the turn routers and the routers that are *HPCmax* apart from the node where the pre-SSR originates.

[3] Owing to contention, the SSR and data flits may not travel same number of hops as the pre-SSR and can be stopped and buffered in an intermediate node.

A. Scalability of the WiSMART NoC

In this section we analyze the scalability of the WiSMART NoC with increasing operating frequencies and system sizes. Fig. 8 compares the zero load latency for uniform random (UR) traffic in a 256-core system incorporating 1D RSMART mesh and WiSMART NoCs, operating at different clock frequencies. The WiSMART NoC consists of four 64-core super-squares. For increasing clock frequencies, the *HPCmax* values of both the NoCs decrease linearly, starting form *HPCmax*=8 for a clock frequency of 1.375GHz. For lower *HPCmax* values (i.e., high frequencies), the data exchanges in a RSMART mesh NoC involve high number of SMART hops (Fig. 2(a)) and hence, exhibit high network latencies (Fig. 8). On the other hand, the WiSMART NoC provides single cycle communication links between the physically far apart nodes, even with high clock frequencies. Thus, for increasing clock frequencies, the WiSMART NoC achieves increasingly higher gains, when compared to the RSMART mesh NoC. More specifically, WiSMART NoC provides 35% and 45% reduction in latency when compared to the 1D RSMART mesh NoC with 2.75 and 3.667 GHz clock frequencies, respectively. This pattern was also observed for different system sizes and is corroborated through Fig. 9 which compares the zero load latency under uniform random traffic for four different system sizes incorporating 1D RSMART and WiSMART NoCs. Fig. 9 also shows the variation in the NoC performances with varying SMART hop latencies, considering both 3 and 4 cycles per SMART hop. It is important to note that with different NoC frequencies, the number of clock cycles required to implement all the four stages of a SMART hop may vary (higher the frequency, higher the number of cycles required to implement the router stages in a SMART hop). As seen from Fig. 9, the gain achieved by using the WiSMART increases with increasing router-stage latency. This further demonstrates the usefulness of the WiSMART mechanism in designing NoCs that operate at high clock frequencies.

B. Flit Latency in the NoC with Real Applications

In this section, we compare the flit latencies of WiSMART, flattened-butterfly high radix (HR-FB) and the 1D RSMART mesh NoCs, considering the above mentioned 10 applications. We consider a 16×16mm die with 64 cores. HR-FB NoC operates at 3GHz and uses 4-stage routers and pipelined long-range wires (up to 3 stages) [12]. To provide a fair comparison with this HR-FB NoC, we use *LPCmax=4mm* (hence a clock frequency of 2.75GHz) for the WiSMART NoC. We consider 2 different 1D RSMART mesh NoCs with *LPCmax* values of *8mm* (1.375GHz clock) and *4mm* (2.75GHz clock), denoted as RSMART8 and RSMART4, respectively.

As observed form Fig. 10, RSMART8 exhibits the highest network latency, mainly due to its low NoC clock frequency. When compared to RSMART8, the RSMART4 mesh NoC achieved a 33% reduction in the flit latency. This once again demonstrates that using a NoC with a high operating frequency and a small *HPCmax* is more beneficial than using a high *HPCmax* NoC that is operating with lower clock frequency.

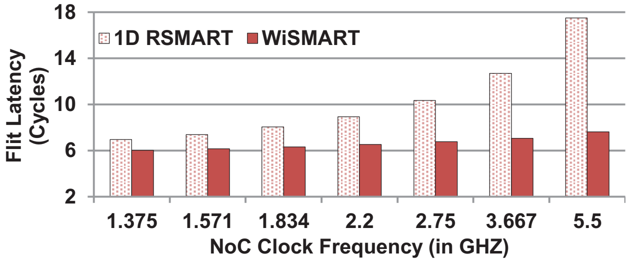


Fig. 8. Zero load latency of 256-core NoCs for different clock frequencies.

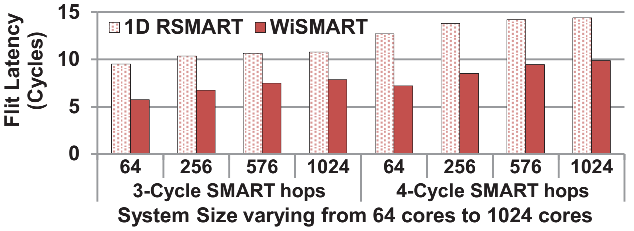


Fig. 9 Zero load latency under uniform random traffic for two different SMART-hop latencies (3 and 4 cycles). Here, *LPCmax* is *4mm* is for all systems. The 64, 256, 576 and 1024 core WiSMART NoCs are made of 4 super-squares each sized 16, 64, 144 and 256 cores, respectively.

In WiSMART and RSMART NoCs, the data flits bypass the router stages at intermediate and destination nodes. However, in HR-FB, the data flit traverses all stages in a router (on average 2.6 router traversals). Also, HR-FB uses latency-expensive routers and links. Using latency-efficient wireless links, WiSMART NoC outperforms both the RSMART4 (on average 3.17 SMART hops) and HR-FB NoCs. On average, WiSMART NoC achieves 57%, 46% and 37% savings in flit latency when compared to RSMART8, HR-FB and RSMART4 NoCs respectively (Fig. 10). Among the considered applications, CNL, RAD, FFT, GP and KM exhibit high fractions of long range traffic injections [9][22]. Hence, these applications benefit more from the use of WiSMART NoC. LR and PCA exhibit mainly short range traffic and hence achieve relatively low latency improvements by employing the WiSMART NoC (26% gain when compared to the RSMART4). Finally, the WIs in the WiSMART NoC also help in efficiently distributing the hotspot-heavy skewed traffic patterns exhibited in WTR and CNL [9]. In these applications, a few hot-spot cores inject high volumes of traffic, causing network congestion. Unlike the wireline-only NoCs that use wireline links for all data transfers, the WiSMART NoC routes a certain percentage of traffic (long-range traffic) through the wireless links that leads to the alleviation of bandwidth bottlenecks, and resulting in improved system performances.

C. Full System Energy Delay Product

In this section, we evaluate the full system energy delay product (EDP) of a 64-core system. As demonstrated earlier, among the four different NoC architectures considered here, WiSMART and RSMART4 perform better than the others. Hence, we consider these two architectures for this EDP

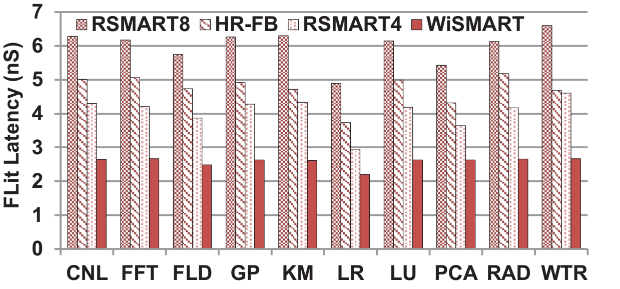


Fig. 10. Average flit latencies for the four different NoC architectures.

analysis. As mentioned in Section IV.B, the considered NoCs operate at 2.75 GHz. For EDP computation, we use the applications runtimes and the total energy consumed by the computing cores and the network components. The core-level statistics generated by the GEM5 simulations are incorporated into McPAT (Multicore Power, Area, and Timing) to determine the core power [25]. The energy consumption of the network components are computed following [3]. The power consumption of the WIs is computed following [8].

Some of the WiSMART NoC routers can dissipate more power than 1D RSMART NoC routers, due to the presence of WIs. However, the share of the power consumed by a WI (which is in order of mW) in total system power dissipation (in order of tens of Watts) is usually negligible. Moreover, by reducing the message transfer latencies, the WiSMART NoC lowers the execution times of the applications and ultimately leads to lower full system energy consumptions. This statement is corroborated through Fig. 11, which compares the full system EDPs of the RSMART4 and WiSMART NoC-based multicore systems for the ten applications. Among the considered applications, GP and CNL exhibit the two highest flit-intensity values (average number of flits per instruction) while FFT and LR have the two lowest flit-intensity values. Hence, CNL and GP benefit most from the better NoC architecture and achieve 22% and 27% EDP gains respectively, with the WiSMART NoC when compared to the RSMART NoC. Despite achieving high latency gains with WiSMART (shown in Fig. 10), FFT and LR achieve only 6% and 7% full system EDP savings mainly due to their low flit-intensity. For the considered applications, the WiSMART NoC achieves an average of 16% of EDP savings when compared to the RSMART NoC.

V. Conclusion

By using wires with clockless repeaters and dedicated router bypass request networks, the SMART NoC architectures aim to establish single cycle data-transfers among the physically far apart on-chip nodes. However, in a SMART NoC, the number of intermediate nodes that can be bypassed within a single NoC clock cycle depends on the operating frequency. On the other hand, the on-chip wireless links are capable of establishing single-cycle data transfers among the physically far apart nodes even with high clock frequencies. In this work, we proposed a wireless communication enabled SMART NoC (WiSMART) that is incorporated with a novel *look-ahead request* based routing mechanism. Our analysis show that WiSMART NoC is more scalable and latency-efficient compared to the wireline SMART NoC. For the considered applications, with a 2.75GHz clock, the WiSMART achieves about 33% reduction in network latency and 16% savings in the full system EDP when compared to a wireline SMART NoC.

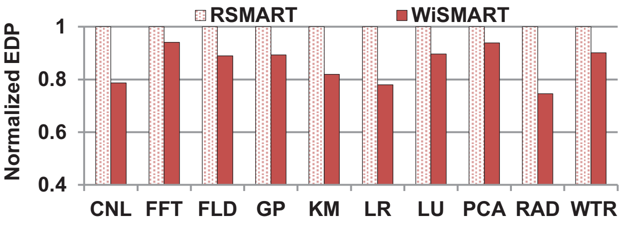


Fig. 11. Comparison of the full system EDP.

References

[1] R. Marculescu, et al. "Outstanding Research Problems in NoC Design: System, Microarchitecture, and Circuit Perspectives," *IEEE Trans. on Computer-Aided Design of Int. Circuits and Systems*, 28(1), pp.3-21.

[2] U. Y. Ogras and R. Marculescu, "It's a small world after all: NoC performance optimization via long-range link insertion." *IEEE Trans. on VLSI*, 14(7), pp.693-706.

[3] T. Krishna et al., "Breaking the on-chip latency barrier using SMART." In *Proc., 19th International Symposium on High Performance Computer Architecture (HPCA 2013),* pp. 378-389.

[4] T. Krishna et al. "Smart: Single-Cycle Multihop Traversals over a Shared Network on Chip," *IEEE micro*, *34*(3), pp.43-56.

[5] X. Chen, N. K. Jha, "Reducing Wire and Energy Overheads of the SMART NoC Using a Setup Request Network". *IEEE Transactions on Very Large Scale Integration (VLSI) Systems,* (available online).

[6] S. Deb, et al. "Wireless NoC as Interconnection Backbone for Multicore Chips: Promises and Challenges." *IEEE Jnl. on Emerging and Selected Topics in Circuits and Systems*, 2(2), 228-39.

[7] A. Karkar, et al. "A Survey of Emerging Interconnects for On-Chip Efficient Multicast and Broadcast in Many-Cores". *IEEE Circuits and Systems Magazine*, 16(1), pp.58-72.

[8] P. Wettin, et al., "Design Space Exploration for Wireless NoCs Incorporating Irregular Network Routing." *IEEE Trans. on Computer-Aided Design of Int. Circuits and Systems*, 33(11), 1732-45.

[9] R. Kim, et al. "Wireless NoC for VFI-Enabled Multicore Chip Design: Performance Evaluation and Design Trade-offs," *IEEE Trans. on Computers*, 65(4), 1323-36.

[10] J. Kim, et al. "Flattened butterfly: a cost-efficient topology for high-radix networks." *ACM-SIGARCH Comp. Archi. News,* 35(2), 126-37.

[11] N. Abeyratne, et al. "Scaling towards kilo-core processors with asymmetric high-radix topologies." In the *Proc., of the 19th IEEE Intl. Symp. on High Performance Computer Architecture*, *2013.*

[12] K. Sewell, et al. "Swizzle-switch networks for many-core systems". *IEEE Jnl. on Emerging & Selected Topics in Cir. and Sys.*, 2(2),278-94.

[13] S. Park, et al. "Approaching the theoretical limits of a mesh NoC with a 16-node chip prototype in 45nm SOI." in *Proc. 0f DAC, 2012*.

[14] A. Kumar, et al. "Toward Ideal On-Chip Communication Using Express Virtual Channels," *IEEE micro*, 28(1), pp.80-90.

[15] T. Krishna, et al. "NoC with Near-Ideal Express Virtual Channels Using Global-Line Communication," In *Proc., HOTI '08.*

[16] V. F. Pavlidis et al. "3-D topologies for networks-on-chip." *IEEE Trans. on Very Large Scale Integration (VLSI) Systems*, 15 (10), 1081-1090.

[17] A. Kumaryet al. "A 4.6Tbits/s 3.6GHz single-cycle NoC router with a novel switch allocator in 65nm CMOS," In *Proc., ICCD-2007.*

[18] Y. Hoskote, et al. "Teraflop prototype processor with 80 cores". In *Proceedings of the Symp. on High Performance Chips*.

[19] K. Duraisamy, et al. "Enhancing Performance of Wireless NoCs with Distributed MAC Protocols", in *Proc., of ISQED*, 2015.

[20] C. Bienia, "Benchmarking Modern Multiprocessors," Ph.D. dissertation, Dept. Comp., Sci., Princeton Univ., NJ, 2011.

[21] S.C. Woo, et. al., "SPLASH-2 Programs: Characterization and Methodological Considerations," In *Proc. of ISC- 95*, pp. 24-36.

[22] K. Duraisamy, et al. "High-Performance and Energy-Efficient Network-on-Chip Architectures for Graph Analytics". *ACM Trans. Embed. Comput. Syst*. 15 (4), Article 66 (September 2016).

[23] J. Talbot. Et al. "Phoenix++: modular MapReduce for shared-memory systems". *Proc., of 2nd intl., workshop on MapReduce & its applications*.

[24] N. Binkert, et al. "The GEM5 Simulator", *ACM SIGARCH Computer Architecture News*, 39(2):1-7.

[25] S. Li, et al., "McPAT: an integrated power, area, and timing modeling framework for multicore and manycore architectures", In *Proc., of the 42nd IEEE/ACM Intl. Symp. on Microarchitecture,* pp. 469-480.