A Case for Low Frequency Single Cycle Multi Hop NoCs for Energy Efficiency and High Performance

Monodeep Kar and Tushar Krishna
School of Electrical and Computer Engineering
Georgia Institute of Technology
Atlanta, Georgia 30332–0250
monodeepkar@gatech.edu, tushar@ece.gatech.edu

***Abstract*—As the number of cores in a multi-core system increase, network on-chip (NoC) latency and transmission energy scale unfavorably, since they are directly proportional to the number of hops traversed. Designers often have to trade-off energy to get lower latency (for instance long-distance bypass links with high-radix multi-stage routers) or latency to get lower energy (e.g., scaling down voltage and frequency of NoC routers and links). This work offers an alternate design-space for latency-energy optimization that has previously been unexplored, by harnessing the fact that lower frequency links can actually be used to transmit over longer on-chip distances within a cycle. We leverage a recently proposed micro-architecture that enables the construction of single-cycle multi-hop paths on the fly over a regular mesh network, and augment it with support for dynamic voltage and frequency scaling by decoupling router frequency from link frequency. In essence, we enable packets to traverse only wires from the source to the destination (as if it had a dedicated connection) only getting buffered at routers if necessary (at turns or due to contention). We address the synchronization challenges of multi-hop bypass setup signals in a multi-frequency domain and propose novel static/dynamic router and link frequency assignment techniques. Across synthetic as well as full-system benchmarks, we demonstrate reduced energy with similar or better run-times.**

I. Introduction

Chip-Multiprocessors (CMP) with more than 100 cores are soon going to become an integral part of Exascale computing and the network-on-chip (NoC) connecting these cores will be critical to the overall performance of the system. Energy-efficiency and latency of NoCs are two key aspects that need to be addressed in such designs to achieve scalability. The biggest scalability challenge for NoC energy is that wire capacitance (hence energy) is an order of magnitude higher than transistor capacitance, meaning that the data movement energy dominates, especially for long distances across the chip [1]. The biggest scalability challenge for NoC latency is that it is directly proportional to the number of hops traversed [2], even with highly-optimized single-cycle routers [3].

Energy consumption has become a first order design metric today with the end of Dennard's scaling. The NoC power already contributes 10-30% [4] of the chip power budget for existing designs and is expected to increase its share with increasing number of cores. Dynamic voltage and frequency scaling (DVFS) is one of the most popular and well-studied techniques for adaptively balancing performance and energy efficiency, and is used extensively in processors today, especially with the integration of fast on-chip voltage regulators. For the same reason, DVFS for NoCs has also garnered a lot of interest recently [5]–[11], [11]–[14]. The key challenge with DVFS though is that it, by definition, trades-off performance (latency, throughput) for energy-efficiency, and is thus used sparingly during moments of low activity.

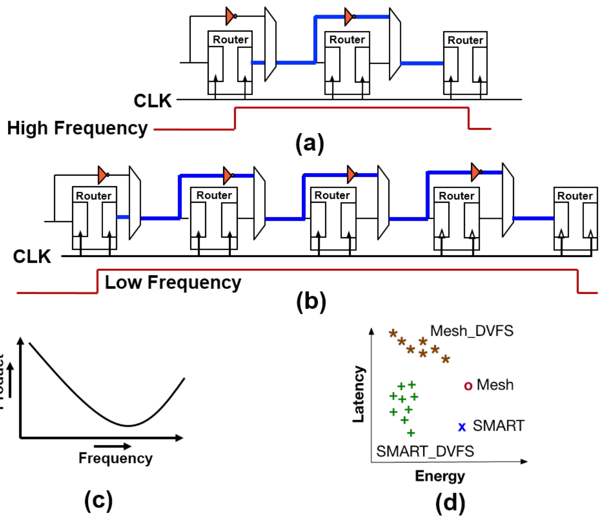

Fig. 1. (a) Example of single-cycle multi-hop traversal in a SMART NoC (b) A SMART NoC with a lower clock frequency. (c) Energy delay product of a SMART network against uniform scaling of frequency (all routers have same frequency). (d) Qualitative comparison of Mesh, SMART, traditional DVFS, and $\text{SMART}_{\text{DVFS}}$ (this work)

To reduce on-chip latency, the fundamental solution is to reduce the number of hops traversed. High-radix topologies [2] add additional links between distant routers to reduce latency by bypassing intermediate routers. The challenge with this approach, though, is that the multi-ported routers from where these express links originate and terminate add huge energy and area overheads [5]. This is because the energy and area of structures such as the crossbar inside the router scale quadratically with the number of ports; not to mention additional buffers and arbitration logic at each new port. More buffers and wires in the NoC also increases leakage [15].

A recent NoC microarchitecture called SMART [16], [17] proposes to address the aforementioned dependence on hops by exploiting the fact that although interconnect scaling has plateaued compared to logic in modern technology nodes, repeated wires are fast enough to transmit across 10+mm at a GHz. Since frequency of NoC routers today and in future will be limited due to the power wall, we can construct single-cycle multi-hop bypass paths across multiple routers. Fig. 1(a) shows an example of a 2-hop path being traversed in one-cycle. The challenge with SMART, however, is that the number of hops that can be bypassed scales down as clock frequency or tile size goes up, limiting its applicability only to domains with small tiles and slow clocks. Moreover, SMART does not directly address the energy challenge of long distance traversal since the same number of routers and links are still traversed as a conventional mesh (albeit in one cycle rather than multiple).

In this work, we leverage the idea of asynchronous bypasses in SMART to introduce a new design-space exploration point to DVFS. Compared to a baseline Mesh, traditional DVFS lowers energy at the cost of performance (latency and/or throughput) while traditional SMART improves performance at the same energy, as Fig. 1(d) shows. What if we could get both? We make the following observation: scaling the frequency (i.e., lower energy) in a SMART network can potentially allow packets to bypass more number of hops in one clock cycle (i.e., higher performance), as Fig. 1(b) demonstrates. This design-space of leveraging frequency scaling to dynamically change the distance traversed within a cycle is ripe for optimization, and to the best of our knowledge has not been explored before. Fig. 1(c) illustrates that there could be a potential sweet-spot providing the lowest EDP with increasing frequency, which in turn can enable design points with both lower latency and lower energy than those afforded by just SMART or DVFS alone (Fig. 1(d)).

This paper presents a methodology to enable DVFS over SMART NoCs. This introduces new challenges not present in conventional NoC DVFS schemes:

- The cycle time for a multi-hop traversal will be longer than that in the baseline design. Moreover, a reduced frequency increases the time spent in each router upon an unsuccessful bypass. Both of these can in fact end up hurting performance of SMART, requiring careful optimization.
- Setting up single-cycle multi-hop bypass paths is non-trivial in a domain where multiple-frequency islands exist, multiple nodes operating at different frequencies may want to setup paths, and there is no unique definition of a cycle.

We address both challenges. We also present a simple policy for dynamic voltage-frequency assignment for the SMART DVFS NoC utilizing the multi-hop bypass requests as a proxy of network traffic. Across a suite of synthetic traffic workloads and full-system PARSEC simulations, we demonstrate the same (or better) performance at lower energy and lower EDP, giving an overall win-win.

II. Background and Related Work

A. Single-Cycle-Multi-Hop Networks

Wire Delay: Single-cycle Multi-hop Asynchronous Repeated Traversal (SMART) NoCs [16], [17] exploit the observation that global repeated wires are fast enough to send signals across 10+ mm within 1ns. SMART NoCs augment mesh routers with a bypass mux (that acts as a repeater) and enable flits to traverse multiple routers asynchronously in one cycle before getting latched, as Fig. 1(a) shows. A flit is the smallest unit of a packet, and equals the link width. The maximum number of hops that can be traversed in a cycle is a design-time parameter known as HPC_{MAX} (maximum hops per cycle), which depends on (a) the underlying repeated wire delay at the particular technology node, (b) the clock frequency, and (c) the tile size. The authors in SMART [17] observed a HPC_{MAX} of 9 to 11 at 45nm at 1GHz with 1mm $\times$ 1mm tiles.

Operation of a SMART NoC:

1) *Cycle 0: Local Switch Allocation (SA-L).* Each router performs arbitration among the locally buffered flits just like a regular mesh router.
2) *Cycle 1: SMART Setup Request (SSR) and Global Switch Allocation (SA-G).* For every winning flit, the router sends a SSR to all the neighboring routers within a HPC_{MAX} neighborhood in the particular output direction (North/South/East/West) the flit wishes to go out from. These requests are sent over a separate set of control wires that span up to HPC_{MAX} hops in each dimension, and are $\log_2(\text{HPC}_{\text{MAX}})$ bits wide. The SSR carries the number of hops that the flit wishes to bypass, up to HPC_{MAX}. All intermediate routers perform arbitration among the incoming SSRs as well as the local winner (which would have sent its own SSR). If any of the SSRs for the flit wishing to bypass this router win the arbitration, the bypass mux is enabled. SSRs are prioritized based on distance, with the local flit getting highest priority and the furthest one the least (known as Prio=Local [17]). This means that in case of SSR contention, the bypassing flit would be stopped (by disabling the bypass mux) and the local flit sent out instead on the output link.
3) *Cycle 2: Single-cycle Multi-hop Traversal.* The flit is sent out from the router and in the best case bypasses all intermediate routers (as Fig. 1(a) and 1(b) show) till the HPC_{MAX} boundary (or destination router). In case of contention, it might get buffered mid-way and re-arbitrate for a multi-hop path in the subsequent cycles.

SMART bypasses are opportunistic, subject to contention. No explicit acknowledgement (ACK) is required. All flits use XY routing. Flits wishing to turn first request bypass paths along the X dimension till the turning router, and then along Y.

B. DVFS in NoC

All the existing works on DVFS on NoCs [5], [8], [9], [12]–[14], [18]–[20] try to perform DVFS on lightly loaded routers to minimize the performance penalty of DVFS. DVFS for NoCs also introduce additional design challenges:

Bi-Synchronous FIFOs: Bi-synchronous FIFOs enable writes and reads at different frequencies, and are a standard modules required for clock-domain crossings at the router interfaces. However, these introduce additional delays.

Multiple Voltage Supply Lines: The existing works on DVFS in NoC assume the use of multiple supply lines for accessing different voltages. However, use of multiple voltage rails requires multiple voltage converters and power distribution

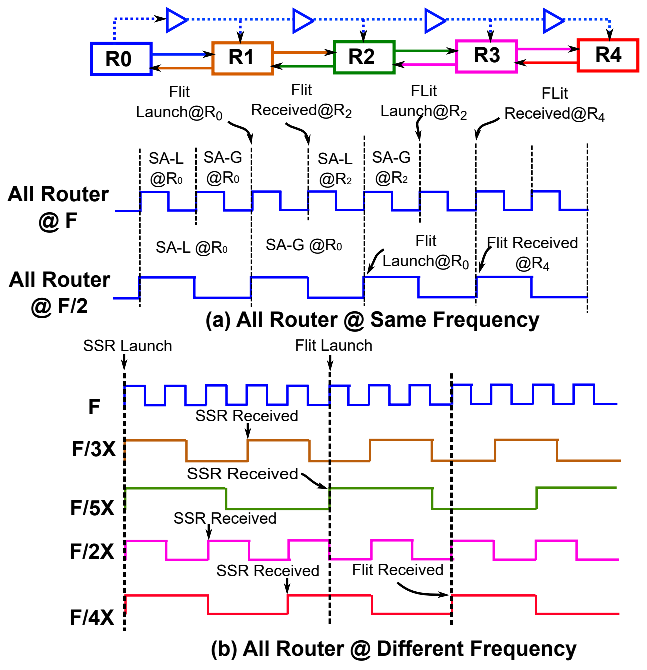

Fig. 2. Synchronization issue of SSR in a SMART router with arbitrary frequency per router.

networks with the area overhead. Our proposed scheme limits the number of unique voltage rails required.

DVFS Assignment Policy: As the router associated with a tile/core serves not only the flits injected from that core, but also those from other cores, the DVFS policy of the NoC fabric has to be different from the one for the core. Prior research on DVFS in NoCs has explored various heuristics for Voltage Frequency Island (VFI) assignment. One set of works use NoC metrics to tune voltage and frequency, such as target throughput [12], [21], buffer utilization [22], energy consumption [19], and errors [23]. Another set uses runtime performance of applications for V-F assignment by observing system-level metrics such as coherence messages [6], L1/L2 misses [7], and memory-access density [9].

III. Motivation and Challenges

A. *Performance implications of higher HPC$_{MAX}$*

For a SMART NoC, the average network latency of a flit is given by the following equation [17]:

$$T = \frac{H}{HPC}t_r + \frac{H}{HPC}t_w + T_C \tag{1}$$

where H is the total number of hops, t_r is the router pipeline delay, t_w is the wire (between two routers) delay, T_C is the contention delay at routers. HPC is the achieved hops per cycle, and can be anywhere from 1 (in case the flit has to stop at every router due to contention) to HPC_MAX (if it successfully bypasses all intermediate routers). A key insight that the SMART paper presents is that most real workloads do not experience heavy link contention since L1 and L2 caches filter most requests into the network [4]. As a result, flits are often able to achieve a high HPC, close to HPC_MAX. Thus a higher HPC_MAX can provide the performance benefits of an all-to-all connected topology, however far the communicating cores may be on chip, via longer bypass paths.

B. *Energy implications of higher HPC$_{MAX}$*

The total energy consumed for sending a flit assuming that it wins both the SA-L and SA-G stages (i.e., no contention) can be represented by the following equation.

$$E = (\frac{H}{HPC}) \cdot [E_{BUF} + E_{SA-L} + HPC_{MAX} \cdot E_{SSR} + HPC \cdot (E_{SA-G} + E_{XBAR} + E_{LINK}) + E_{BUF}] \tag{2}$$

where the first term represents the average number of multi-hop traversals for a flit while the second term represents the energy spent for each multi-hop traversal. Notice that a multi-hop traversal only needs to pay buffering costs at the end points. We can clearly see that as the HPC increases, the energy per flit reduces. This in turn makes a case for increasing HPC$_\mathrm{MAX}$ for lowering energy.

C. *Implications of lower frequency in SMART*

Sections III-A and III-B motivate the benefit of higher HPC$_\mathrm{MAX}$ in SMART NoCs. Recall that HPC_MAX is the maximum hops per cycle, and is directly dependent on tile size and operating frequency. Tile sizes and underlying wire delay are design-time and technology parameters which cannot be changed at runtime. One possible way of achieving higher HPC$_\mathrm{MAX}$ is to lower the NoC clock frequency. Let us examine the performance and energy implications of such a design. In Fig. 2, Router R$_0$ is sending a flit to R$_4$. Assume that there is no other contending flit. At a NoC frequency of F, suppose the HPC$_\mathrm{MAX}$ is two. The flit thus has to stop at R$_2$. The timeline for this traversal is shown in Fig. 2(a). The total traversal takes 7 cycles at frequency F. Now suppose we lower the NoC frequency to F/2. The HPC$_\mathrm{MAX}$ becomes four. R$_0$ can directly setup a SMART path till R$_4$ without stopping at R$_2$. The timeline for this traversal is also shown. The latency for this traversal is 4 cycles at frequency F/2, which is 8 cycles in terms of F. The reason for the corresponding increased delay in each router due to the larger clock period. From an energy point of view, this design point can provide a quadratic reduction as the supply voltage in the routers can be lowered.

This example demonstrates that in SMART, the performance penalty of lowering frequency is much lower than a baseline mesh, where halving the frequency would have doubled the latency. This enables us to get a lower energy point at close to the same network performance, making it a valuable design point for the DVFS controller, enabling it to scale down frequency more aggressively than it can in a traditional design.

D. *Synchronization of SSRs*

If conventional DVFS, as presented by prior works [5], [6], [9]–[11], [23], is applied to a SMART NoC, each individual router can potentially operate at a different voltage-frequency level. In a conventional NoC, such a scenario involves clock-domain crossings at every router through bi-synchronous FIFOs. How would this translate to a SMART NoC with multi-hop paths? On the datapath, a flit can asynchronously pass through multiple routers on the bypass path; a bi-synchronous

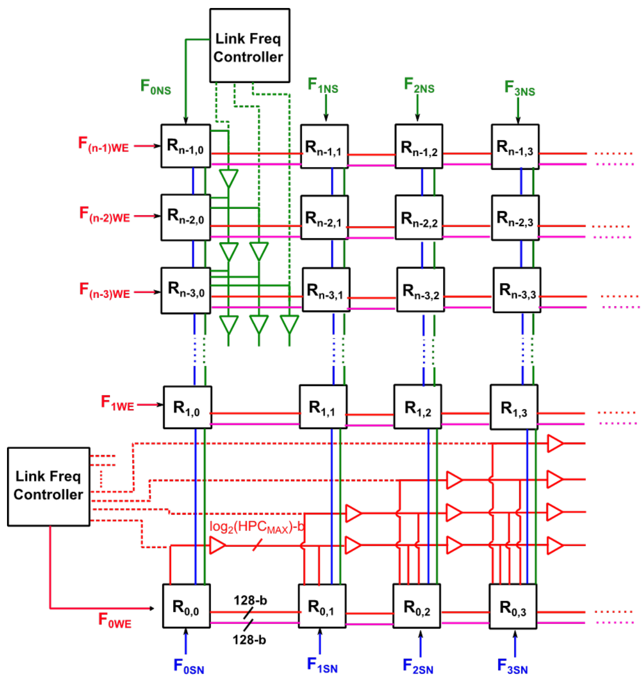

Fig. 3. Architecture of proposed Single-cycle Multi-hop DVFS NoC with decoupled router frequency and per-direction link frequency

FIFO is only needed at the router where the flit stops. The control path however introduces a design challenge. SMART requires all routers along the HPC_{MAX} path to arbitrate during SA-G and setup their bypass muxes accordingly. If every router operates at a different frequency, the bypass muxes may not get set correctly before the flit starts its traversal. Fig. 2(b) illustrates this. It shows five routers in a 1D SMART network, where each router is operating at a different frequency. A SSR is launched by R_0 at a frequency of F. Notice that R_2 has the lowest frequency (F/5X). R_0 needs to wait till R_2 completes its SA-G calculation and sets up its bypass mux before it can launch the flit, as Fig. 2(b) shows. In other words, R_0 has to wait at least for a period of $1/F_{MIN}$, where F_{MIN} is the minimum frequency among the routers within HPC_{MAX} distance of the source router, to ensure that the SSR is 'seen' by all the routers. This synchronization issue limits the throughput of the path to that of the lowest router frequency in the bypass path. Moreover, since SMART does not send any explicit ACKs, it is not obvious how R_0 would know many of its cycles to wait before sending the data flit. Our proposed microarchitecture addresses these issues.

IV. Proposed Network Architecture

A. Control path

In coherence with existing works on DVFS on NoC, we have considered two scenarios, i) where the V-F states of every router change together and ii) individual/a cluster of routers have its own V-F states. Both these are described next.

Uniform VF scaling: The V-F state of every router in the SMART network is same and determined by a centralized DVFS controller. In this case, as all routers operate at the same frequency, no issues arise due to synchronization of SSR signals. The DVFS policy can either aggregate standard network performance metrics like buffer utilization or request-response delay over an epoch, or SSRs as a metric representing the network traffic, as we discuss in Section V-C.

Per direction VF scaling: Section III-C demonstrated that frequency scaling in SMART can increase the waiting time of flits inside routers. Section III-D highlighted that the frequency of a multi-hop path is limited by the slowest clock on that path. Taking these observations into account, we propose a DVFS enabled SMART NoC with the following properties.

1) We decouple router frequencies from link frequencies. Local arbitration takes place at the router frequency, while multi-hop traversal takes place at the link frequency. This provides a new tuning knob to control the HPC_{MAX} while keeping the router wait-time low. Moreover, router supply voltage can be different than the link driver and receiver voltage. We find that lowering router voltage when its frequency is lowered reduces energy, while keeping link voltages high when link frequencies are lowered enables higher HPC_{MAX} values. Together this helps us optimize for both performance and energy, rather than trade one off for the other.
2) We propose to use unique link frequencies for all links along a direction (North/South/East/West) in each row and column of the SMART NoC. The proposed top level architecture is shown in Fig 3. Apart from the local injection/ejection ports, each router has 4 input and 4 output ports and the frequencies of 4 direction of traversals are denoted as F_{WE}, F_{EW}, F_{NS} and F_{SN}. The router frequency is denoted a F_R. For example, in any row, all the flits traversing from west to east will synchronized with respect F_{WE}, different from F_R. Same frequency along each direction ensures that the SSRs do not have the synchronization issue discussed in Section III-D. If a flit wants to turn, it has to stop and cannot bypass through the router. Each row and column has two frequency controllers that set the clock for two traversal directions. Section V-C discusses how the frequencies of the individual directions are determined.

Unlike SMART, where HPC_{MAX} is a design-time parameter, in our design HPC_{MAX} is a runtime parameter that depends on the link frequencies. To account for the maximum possible, SSR wires span the entire row or column in all directions. The width of each SSR wire is $\log_2(k)$ for a $k \times k$ mesh.

B. Micro-Architecture of proposed DVFS SMART router

The microarchitecture of our router is shown in Fig. 4.

Input Port (VC Buffers and Arbiter): We add a level shifter to each input port due to potential difference in voltage between the router and the link. If an incoming flit cannot bypass the router (based on the result of its SSR arbitration), it is latched at the input buffers. The input buffers are bi-synchronous FIFOs where the data is written at the link frequencies and read out at the F_R. The input arbiter operates at F_R and selects a winner from among the Virtual Channels (VCs) of the corresponding input port. Credit signals are sent at F_R and do not need to be re-synchronized.

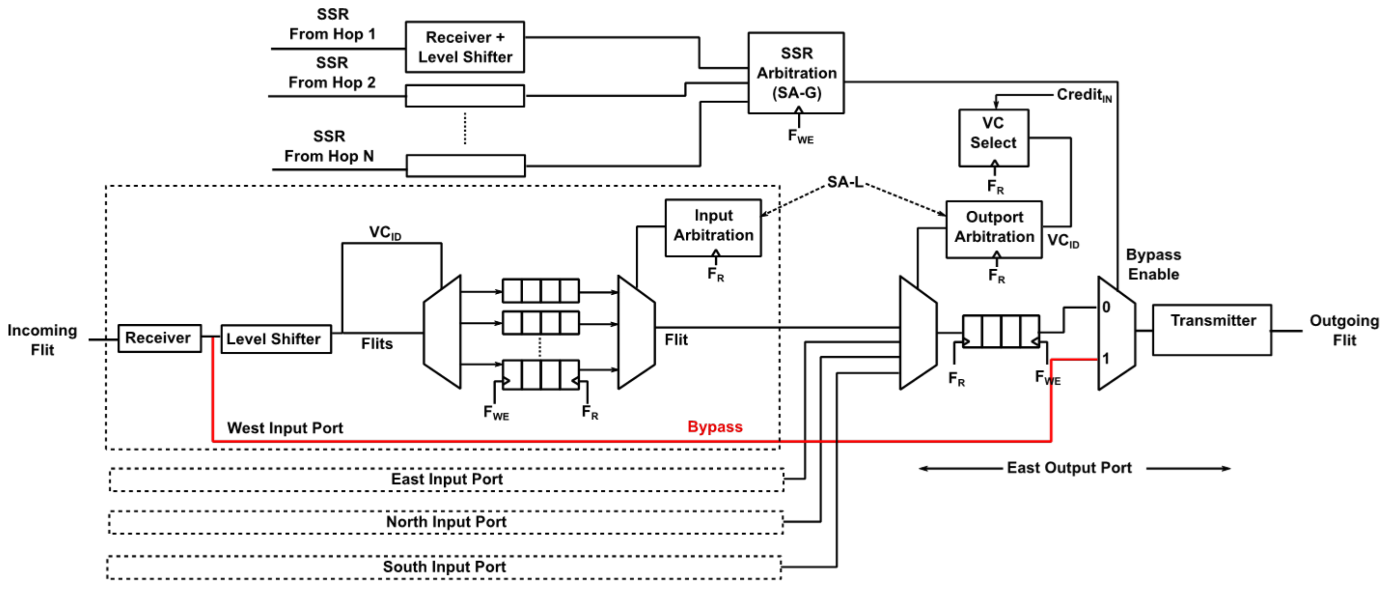

Fig. 4. Micro-Architecture of proposed DVFS router supporting SMART bypasses. The operating frequencies (F_R vs. F_{WE}) for each module are shown.

Output Port (Arbiter, Crossbar, Buffer): The output mux (inside the crossbar) is controlled by the output arbiter logic also operating at F_R and steers the winning input flit to the corresponding output ports. Although the inputs flits from different directions are traversing at different link frequencies, they are internally synchronized with respect to F_R before being steered to the output port. The flit at the output port is written at F_R into a bi-synchronous FIFO and read out at frequency of the corresponding direction.

Bypass Path: The arbiter for the SSRs (i.e., SA-G) operates at the corresponding link frequency and is also interfaced with a level shifter. Each router has dedicated SSR arbiter per direction. The output of the SSR arbiter drives the mux at the corresponding output port. If the incoming SSR wins the arbitration, the select line of the bypass mux is made high at the next link cycle so that the flit can bypass.

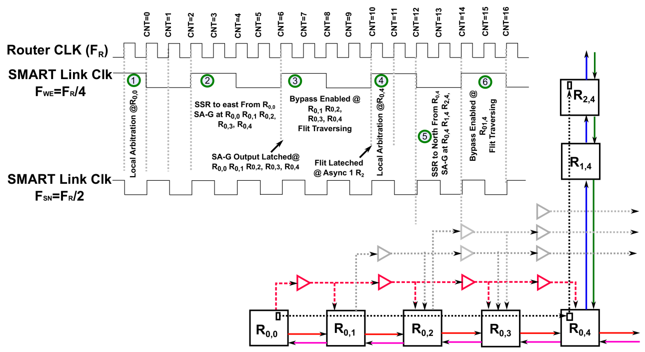
CNT=0
CNT=1
CNT=2
CNT=3
CNT=4
CNT=5
CNT=6
CNT=7
CNT=8
CNT=9
CNT=10
CNT=11
CNT=12
CNT=13
CNT=14
CNT=15
CNT=16
Router CLK (F_R)
SMART Link Clk F_WE=F_R/4
1
Local Arbitration @R0,0
2
SSR to east From R0,0 SA-G at R0,0 R0,1 R0,2, R0,3, R0,4
SA-G Output Latched@ R0,0 R0,1 R0,2, R0,3, R0,4
3
Bypass Enabled @ R0,1 R0,2, R0,3, R0,4 Flit Traversing
Flit Latected @ Async 1 R2
4
Local Arbitration @R0,4
5
SSR to North From R0,4 SA-G at R0,4 R1,4 R2,4,
6
Bypass Enabled @ R01,4 Flit Traversing
SMART Link Clk F_SN=F_R/2
R2,4
R1,4
R0,0
R0,1
R0,2
R0,3
R0,4

Fig. 5. Cycle by cycle activity for flit traversing from $R_{0,0}$ to $R_{0,4}$ followed by $R_{0,4}$ to $R_{2,4}$ through the proposed NoC.

C. Example Operation

We demonstrate the operation of the SMART DVFS NoC using examples. In Fig. 5, router $R_{0,0}$ wants to send a flit to router $R_{2,4}$. Let us assume that all routers are operating at F_R, the frequency for west to east direction (i.e., F_{WE}) on Row 0 is $F_R/4$, and the frequency from south to north (i.e., F_{SN}) on Column 4 is $F_R/2$. For simplicity, assume no contention. Fig. 5 shows the operations that take place during each "cycle" of the router and link clocks for this traversal.

1) The flits at $R_{0,0}$ performs local arbitration (SA-L) at F_R.
2) At the rising edge of the next link cycle (F_{WE}), the winning flit sends a SSR to the east direction. The SSR performs global arbitration (SA-G) at routers $R_{0,0}$, $R_{0,1}$, $R_{0,2}$, $R_{0,3}$ and $R_{0,4}$ following Prio=Local. At $R_{0,0}$, this flit wins as it is the local flit. At $R_{0,1}$, $R_{0,2}$, and $R_{0,3}$, the bypass mux is set. If there was a local contending flit at $R_{0,1}$, $R_{0,2}$ or $R_{0,3}$, the bypass mux would be disabled due to Prio=Local.
3) At the rising edge of the next link cycle (F_{WE}), the flit is sent out and it performs a 4-hop traversal, bypassing all the intermediate muxes and crossbars. The flit is latched at $R_{0,4}$ at F_{WE} and goes through the bi-synchronous FIFO.
4) The flit performs local arbitration (SA-L) at $R_{0,4}$ at F_R. The router frequencies at $R_{0,4}$ need not be same as that at $R_{0,0}$ for the design to work.
5) At the rising edge of the next link cycle (F_{SN}), the flit sends a SSR north. This SSR enables the bypass mux at $R_{1,4}$.
6) The flit performs a 2-hop traversal till $R_{2,4}$ at the rising edge of F_{SN} and gets latched at the next rising edge.

V. Implementation

A. Circuit Implementation

To reduce the transmission energy on the link, we leverage a low-voltage single-ended signaling on the links between the routers [16]. The circuits at the end points of our links are shown in Fig. 4. The Tx operates at the link voltage V_L and uses a voltage-locked repeater circuit [16]. The Rx converts the low-swing signal back to V_L. In case of a bypass, this signal is directly forwarded to the output port (Fig. 4), else it goes through a level-shifter to transfer it to the router voltage V_R and go to the bi-synchronous input FIFO.

B. Clock Distribution and Frequency Generation

We assume a global clock is distributed throughout the entire chip and is used as the router clock. To generalize the proposed microarchitecture, we have used a bi-synchronous FIFO for each of the incoming and outgoing port. However as multiple works have reported, frequency scaling with scaling factor of power of 2 significantly simplifies the design and verification of the asynchronous FIFOs [10]. Therefore, the link frequencies are derived from the router frequency using power of 2 (F_R, $F_R/2$, $F_R/4$). As the link frequencies are locally generated, following design simplifications are achieved.

- Timing margin is usually quite tight with bi-synchronous FIFOs, if the clock domains are asynchronous. However in this case as the link clocks are synchronously derived FIFO design is simplified. Similarly timing closure through synthesis and place-and-route becomes simpler.
- For clock distribution, we use a two-bit frequency id (F_{ID}) per direction per row/column, which is sent to every router. For example, F_{1WE} determines the link frequency of the west to east links in row 1 (Fig. 3) and is distributed across all the routers in row 1, which locally generates the corresponding clock for launching the data across the link (F_{WE}).

C. Frequency Controller

Router vs. Link Frequency: Our architecture requires the same link frequency across the entire direction in each row and column. Each row and column can independently set their own frequencies. This can be done statically or dynamically. Each router can operate independently at its own frequency F_R and the design will be functionally correct without any synchronization issues. However, if F_R is lower than the frequency of any of its outgoing links, the effective link frequency will become limited by F_R. This is because local arbitration (SA-L) occurs at F_R. Thus we recommend setting $F_R \geq \max(F_{WE}, F_{EW}, F_{NS}, F_{SN})$.

Per-Direction Link Frequency: We provide a unique control knob to the NoC, not available in traditional designs, where we can change the frequency of individual rows/columns and get lower energy points for the same performance. In prior works, the policy of assigning voltage-frequency (V_F) states typically uses accessible network metrics as discussed in Section II-B. In a SMART NoC, the network congestion can also be estimated by observing the total number of SSRs sent by the routers. We propose to use the structure shown in Fig. 3 for estimating the traffic in each direction in each row/column. The valid bit of all SSRs spanning a direction enter the Link Frequency Controller (LFC). For every SSR that is initiated, an accumulator in its corresponding LFC is incremented by one. Over a configurable time-epoch, the LFC uses the accumulator count to determine if the link frequency in that direction needs to change. A unique design-point in SMART DVFS is whether higher SSR activity should lower or raise the frequency, since that trades off HPC_{MAX} versus link throughput. We experiment with both design points in our evaluations.

VI. Evaluation

We use the gem5 + Garnet [24] infrastructure for all our evaluations, which provides a cycle-accurate NoC timing model. Network energy is calculated using DSENT [25] where we model components corresponding to SMART and our additions. The energy of the low-swing transmitter-link-receiver is estimated from chip measurements [16].

Target System: For synthetic traffic, we model a 256-core system. Full-system runs use a 64-core system. We model a 32nm technology node, and choose a clock frequency of F=2GHz. We observe a baseline HPC_{MAX} of 4 at this configuration [17] which we validate via DSENT. We use the following (V,F) configs: (1V, F), (0.9V, F/2), (0.75V, F/4).

System Configuration: We use the following naming scheme: MESH-FX is a regular mesh with X as a factor by which Router frequency (F_R) is scaled down; SMART-RXLY is a SMART topology with F_R scaled down by X and all the link frequencies (for every direction) are scaled down by Y; SMART-R1Dyn is a SMART with routers operating at the highest frequency and per-row-column per-direction frequencies set by our LFC (Section V-C).

A. Synthetic Traffic

Performance: The proposed network is first evaluated against synthetic traffic patterns. The results for bit-complement traffic are shown in Fig. 6. First the MESH-FX systems are compared with SMART-RXLX systems, i.e. SMART systems with routers and links assigned to the same frequency. The baseline SMART system (SMART-R1L1) achieves lower low-load-latency than MESH-F1, as already demonstrated before [17]. However uniformly reducing frequencies both in SMART and MESH behaves differently. While in MESH-F2, the low-load latency is doubled and the network throughput is almost halved, SMART-F2 resulted in an improved throughput than MESH-F2 with a low-load latency of 17.68, which is still lower than the low-load-latency of 22 cycles for MESH-F1. This clearly demonstrates the potential of adjusting router and link frequency uniformly while maintaining an acceptable network throughput. At ultra-low injection rate, a SMART-R4L4 will have similar latency with MESH-R1L1 with a significant amount of energy benefit. We note that uniformly scaling router and link frequency works seamlessly with the baseline SMART design [17].

Next, taking advantage of our proposed router micro-architecture, we evaluate SMART-R1L2 and SMART-R1L4. SMART-R1L2 achieves a lower load-load latency and SMART-R1L4 achieves a similar latency compared to SMART-R1L1 which is attributed due to higher hops per cycles, and reduced time spent at the intermediate routers compared to SMART-R2L2 and SMART-R4L4. The saturation throughput of SMART-R1L2 and SMART-R2L2 are similar. This is attributed to the fact that the router can only send packets every two cycles for both these networks, therefore at sufficiently high injection rate, backpressure builds up at NIC-router buffer and limits the throughput.

Uniform vs. Per Row/Column Frequency Allocation: From the previous observation, one can see that reduced link frequency helps in lowering low-load latency, however network throughput remains unaffected. The most congestion in a Mesh happens typically at the middle of the network

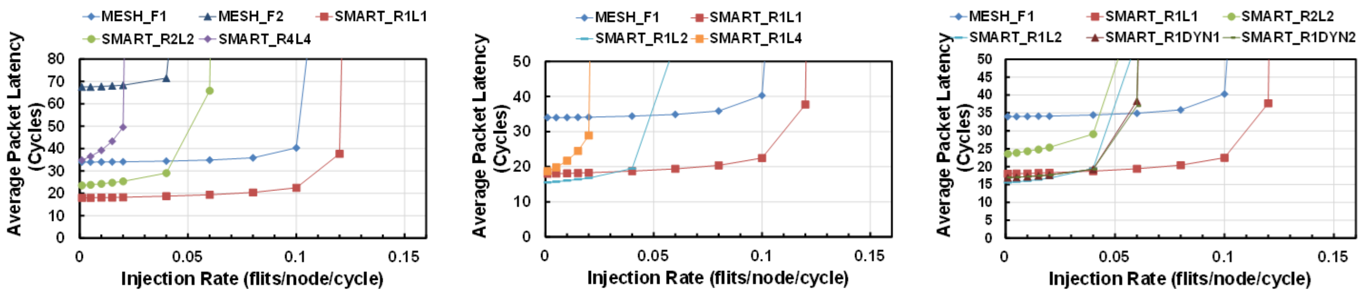

Fig. 6. Performance of proposed network with bit-complement synthetic traffic for different flit-injection rate (System: 16x16 Mesh with XY routing).

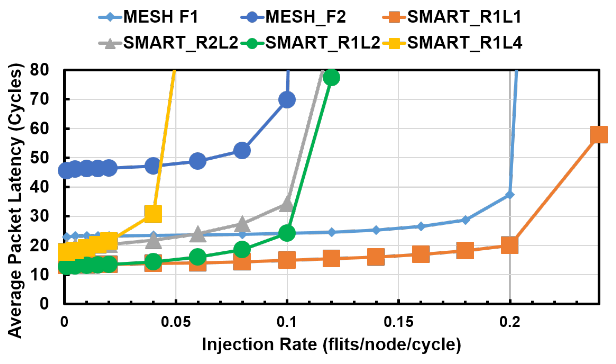

Fig. 7. Performance with uniform random traffic in a 16x16 mesh

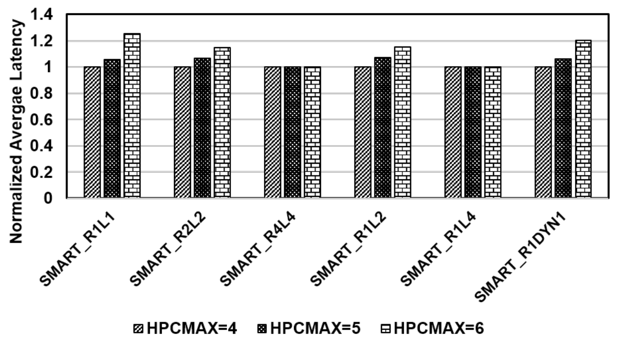

Fig. 8. Impact of HPC_{MAX} on network delay (at low injection rate, bit-complement synthetic traffic, 16x16 mesh)

for most synthetic traffic patterns [18]. To improve the network saturation rate, our link frequency controller assigns the highest link frequency to the links crossing the center of the network whereas the links near the periphery of the network operate at lower frequency. The average latency plot shows that the low-load-latency of SMART-R1LDYN is similar to SMART-R1L1 and SMART-R1L2. The worst case latency across a network for bit-complement traffic is along the periphery of the network which remains unaffected. However, we observe an improvement in the latency near the saturation rate, due to higher link frequencies at the center. The final saturation rate still remains the same. Fig. 7 shows the network performance for a uniform random traffic. The injection rate of network saturation increases uniformly for each configuration, however the trend between the configurations remains same. Fig. 8 shows the improvement in average network latency for the different configurations as HPC_{MAX} is increased from 4 to 5 and 6. SMART-R1L1 and SMART-R1L-DYN has

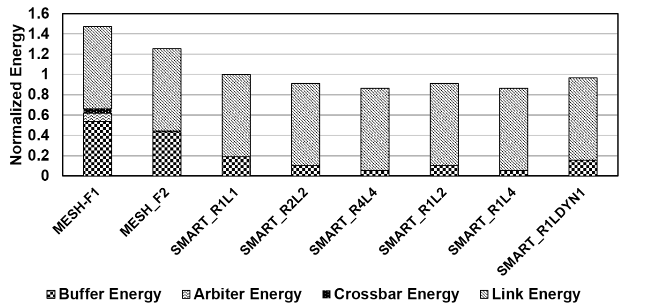

Fig. 9. Total Network Dynamic Energy (at low injection rate, bit-complement synthetic traffic, 16x16 mesh)

the maximum improvement, as links operate at the highest frequency with the least HPC. The improvement reduces for SMART-R1L2 and SMART-R2L2. All SMART configurations with link frequency $F_R/4$ do not show any improvement as these configuration has a HPC_{MAX}=16 due to 4X lower frequency and increasing HPC does not improve throughput for a 16x16 Mesh as only one dimensional bypass is enabled.

Energy: Fig. 9 shows the normalized (with respect to SMART-R1L1) energy breakdown of the system. A baseline MESH (MESH-F1) design has higher energy consumption, than SMART-R1L1. Link energy remains same for all configurations as under all bypass schemes, the flits passes through the same number of links for a given routing scheme. For SMART-R2L2 and SMART-R4L4, the router energy reduces both due to voltage scaling as well as reduced router activity for higher HPC_{MAX}. Router energy also reduces for SMART-R1L2 and SMART-R1L4 (no voltage scaling applied) as buffer energy, which is the most significant fraction of the router energy reduces due to more number of flits being able to bypass the intermediate routers due to a higher HPC_{MAX}. SMART-R1LDyn shows higher energy as the frequency selection algorithm assigns link frequencies from either F_R or $F_R/2$, however due to low injection rate, the dynamic frequency selection does not help network congestion. Fig. 10 shows the normalized (with respect to SMART-R1L1) delay-vs-energy plot for different configurations. Clear trends can be seen here. Traditional DVFS (MESH-F2) lowers energy at the cost of delay. Uniform frequency scaling associated with router voltage scaling (SMART-R2L2 and SMART-R4L4) improves energy, however increases delay. For our proposed router and link frequency decoupled schemes (SMART-R1L2 and SMART-R1L4), a lower network latency with lower energy is achieved demonstrating the benefit of our proposed methodology.

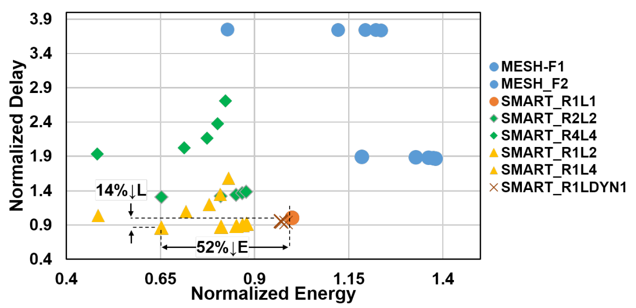

Fig. 10. Energy Delay plot for the proposed network for different configurations for multiple injection rates ≤ 0.02 (bit-complement synthetic traffic, 16x16 mesh)

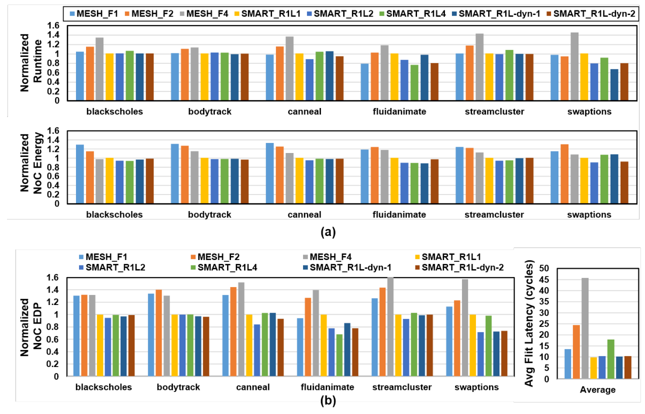

Fig. 11. (a) Application Runtime, Energy, and (b) EDP and average flit delay plot with full system PARSEC workloads (64 in-order SPARC processor)

B. Full System

We perform a full-system simulation over a MOESI directory protocol with a Private L1 and Private L2 per tile. We run the parallel sections of PARSEC [7] for the proposed system. Each run consists of 64 threads of the application running on our CMP. Fig. 11 shows the application runtime and energy (normalized with respect to SMART-R1L1). Frequency scaling in regular MESH increases latency at the cost of reducing energy. SMART-R1L2 and SMART-R1L4, lowers the average runtime by 7.6% and 2.3% and lowers the network energy by 6.3% and 3% respectively. The dynamic link frequency maps are obtained from the frequency selection algorithm. Two configurations with dynamic link frequencies are created from the SSR counts: 1) the rows/columns with higher SSR requests over an epoch are assigned a higher link frequency (SMART-R1L-DYN1) and 2) the rows/columns with higher SSR requests over an epoch are assigned a lower link frequency (SMART-R1L-DYN2). SMART-R1L-DYN2 performs better with an average of 7.8% lower network latency and 2.88% lower network energy. Fig. 11 shows the network EDP for the benchmarks. SMART-R1L2 and SMART-R1L-DYN-2 and 13.28% and 10% improvement in EDP compared to baseline SMART-R1L1 which itself is 21% lower in EDP compared to baseline MESH-F1. Fig. 11 also demonstrates the network latency across the configurations on average across all workloads. We can again see that the network latency with traditional DVFS increases by 2-4X, while with our proposed SMART DVFS schemes it remains fairly constant all schemes, enabling lower-energy designs at the same performance.

VII. Conclusion

In this work we make a case for running single-cycle multi-hop NoCs at lower link frequencies than the rest of the subsystem to enable flits to traverse chip dimensions within one clock cycle. We demonstrate an architecture that provides energy-efficiency with same (or better) overall performance, unlike traditional DVFS schemes that need to trade-off latency against energy. This work opens up a novel design-space of tuning traversal distance with clock frequency and can pave the way for research in energy-efficient high-performance NoCs and DVFS policies for the dark silicon era.

References

[1] S. W. Keckler *et al.*, "Gpus and the future of parallel computing," *IEEE Micro*, 2011.
[2] J. Kim *et al.*, "Flattened butterfly topology for on-chip networks," in *IEEE Micro*, 2007.
[3] S. Park *et al.*, "Approaching the theoretical limits of a mesh noc with a 16-node chip prototype in 45nm soi," in *DAC*, 2012.
[4] B. K. Daya *et al.*, "Scorpio: a 36-core research chip demonstrating snoopy coherence on a scalable mesh noc with in-network ordering," *ISCA*, 2014.
[5] U. Y. Ogras *et al.*, "Voltage-frequency island partitioning for gals-based networks-on-chip," in *DAC*, 2007.
[6] R. Hesse and N. E. Jerger, "Improving dvfs in nocs with coherence prediction," in *NOCS*, 2015.
[7] Y. Yao and Z. Lu, "Dvfs for nocs in cmps: A thread voting approach," in *HPCA*, 2016.
[8] J. Zhan *et al.*, "Optimizing the noc slack through voltage and frequency scaling in hard real-time embedded systems," *TCAD*, 2014.
[9] Y. Yao and Z. Lu, "Memory-access aware dvfs for network-on-chip in cmps," in *DATE*, 2016.
[10] H. Bokhari *et al.*, "Malleable noc: Dark silicon inspired adaptable network-on-chip," in *DATE*, 2015.
[11] X. Chen *et al.*, "In-network monitoring and control policy for dvfs of cmp networks-on-chip and last level caches," *TODAES*, 2013.
[12] X. C. et al., "Dynamic voltage and frequency scaling for shared resources in multicore processor designs," in *DAC*, 2013.
[13] R. David *et al.*, "Dynamic power management of voltage-frequency island partitioned networks-on-chip using intel's single-chip cloud computer," in *NOCS*, 2011.
[14] P. Bogdan *et al.*, "Dynamic power management for multidomain system-on-chip platforms: an optimal control approach," *TODAES*, 2013.
[15] L. Chen *et al.*, "Power punch: Towards non-blocking power-gating of noc routers," in *HPCA*, 2015.
[16] C.-H. O. Chen *et al.*, "Smart: a single-cycle reconfigurable noc for soc applications," in *DATE*, 2013.
[17] T. Krishna *et al.*, "Breaking the on-chip latency barrier using smart," in *HPCA*, 2013.
[18] A. K. Mishra *et al.*, "A heterogeneous multiple network-on-chip design: an application-aware approach," in *DAC*, 2013.
[19] U. Y. Ogras *et al.*, "Design and management of voltage-frequency island partitioned networks-on-chip," *TVLSI*, 2009.
[20] S. Garg *et al.*, "Custom feedback control: enabling truly scalable on-chip power management for mpsocs," in *ISLPED*, 2010.
[21] M. R. Casu *et al.*, "Rate-based vs delay-based control for dvfs in noc," in *DATE*, 2015.
[22] A. K. Mishra *et al.*, "A case for dynamic frequency tuning in on-chip networks," in *MICRO*, 2009.
[23] A. Ansari *et al.*, "Tangle: Route-oriented dynamic voltage minimization for variation-afflicted, energy-efficient on-chip networks," in *HPCA*, 2014.
[24] N. Agarwal *et al.*, "Garnet: A detailed on-chip network model inside a full-system simulator," in *ISPASS*, 2009.
[25] C. Sun *et al.*, "Dsent-a tool connecting emerging photonics with electronics for opto-electronic networks-on-chip modeling," in *NOCS*, 2012.