Rate-based vs Delay-based Control for DVFS in NoC

Mario R. Casu and Paolo Giaccone
Department of Electronics and Telecommunications, Politecnico di Torino, Italy

***Abstract*—Minimization of power via DVFS in an NoC is possible, but may result in an intolerable increase of network delay. We examined two DVFS policies, a *rate-based* policy that scales down frequency and voltage to the minimum value that allows to sustain the injection rate without reaching saturation, and a *delay-based* policy in which a closed-loop control tunes frequency and voltage such that the NoC average delay tracks a target value. We evaluated the power-delay trade-off by means of network simulations and accurate power estimations after synthesis on a 28-nm FDSOI CMOS technology. Our results over synthetic and multimedia traffic patterns show that the first policy largely pays the better saving in power (20%-50% less than the second policy) with a large network delay increase (up to $3\times$). We then conclude that the delay-based policy offers a better power-delay trade-off.**

I. Introduction

Dynamic Voltage and Frequency Scaling (DVFS) is a very effective method for reducing power consumption in CMOS chips. DVFS has been typically used chip-wise to control global voltage and frequency. Recently, its application to individual cores and separately to the NoC has been proposed, among others, by Intel [1][2].

Researchers have proposed to extend such local use of DVFS to individual NoC routers [3][4][5] and links [6]. We focus instead on a *global* application of DVFS to the entire NoC to reduce the overhead of locally replicated DVFS controllers and dc-dc converters, and to remove the resynchronization at the crossing between multiple frequency domains. We assume, however, that NoC and processing elements connected to it have separate DVFS controllers.

We focus only on an NoC with global DVFS and our goal is to understand the achievable power-performance trade-off. This research has been motivated by our observation that a very aggressive DVFS policy leads to a large increase of delay in delivering packets. In relative terms, such increase is significantly more than the power reduction. To illustrate this trade-off we compare two possible DVFS policies. The first policy is more aggressive because it scales the NoC frequency—and so also the voltage—down to the minimum value that allows to sustain the injection rate without reaching saturation. We refer to this policy as *Rate-based Max Slow Down (RMSD)*. The second one sets instead a target delay and throttles the NoC frequency and the voltage to the minimum value that still allows to respect the delay constraint. We call this second policy *Delay-based Max Slow Down (DMSD)*.

To explore the power-delay trade-off under these two policies, we ran NoC simulations with both synthetic and multimedia traffic. To estimate the NoC performance under DVFS, we built a modified version of the Booksim cycle-accurate simulator [7]. We performed logic synthesis and transistor-level simulations of the virtual-channel router used in network simulations to accurately evaluate power consumption as a function of the scaled voltage and frequency, of the injection rate, and of the activity inside the routers and in the links. We targeted for this evaluation a 28-nm FDSOI CMOS technology.

In summary, we provide the following novel contributions:

- We show that a rate-based DVFS strategy like RMSD, which aims at minimizing power consumption with the only objective of sustaining the NoC throughput, pays a very high cost in terms of NoC delay. Moreover, with this DVFS strategy the *delay* expressed in actual time units—not the latency in clock cycles—as a function of the injection rate exhibits an anomalous non-monotonic behavior, which we report for the first time in NoCs.
- A better trade-off between power and delay is obtained with a delay-based DVFS policy. The newly introduced DMSD policy saves less power than the RMSD policy (20%-50%), but it reduces delay substantially (up to $3\times$). It does so with a proportional-integral feedback loop that determines the clock frequency to apply such that the average NoC delay tracks a target delay.
- We show that the DMSD policy produces a better power-delay trade-off no matter the type of traffic (either synthetic or multimedia). The results are also insensitive to router micro-architectural variations, as well as variations of NoC parameters such as packet size and mesh size.

The paper is organized as follows. The related work is presented in Sec. II. The RMSD and DMSD policies are introduced and discussed in Sec. III and Sec. IV, respectively. The experimental results with synthetic traffic are reported in Sec. V, whereas results with multimedia traffic are in Sec. VI. Finally, we draw our conclusions in Sec. VII.

II. Related Work

Previous work in DVFS for NoC focused mostly on DVFS in individual routers or in links [3][4][6]. This method requires costly individual dc-dc conversion (per-link or per-router) and resynchronization at the crossing between frequency domains, which has a detrimental impact on latency. We focus instead on a more realistic scenario in which the NoC is a single voltage-frequency domain, as for instance in [1] and [2].

In processor cores, DVFS controllers minimize power by setting the minimum clock frequency (and so the voltage) that makes the core deliver a given throughput. The controller monitors the status of a workload queue and throttles the speed so that the queue never fills up (core is too slow) or gets empty (too fast) [8]. In NoCs this method can be used by monitoring the buffers located in the routers [5], or the queues located at the crossing of different voltage-frequency islands [9].

An alternative explored by Liang and Jantsch consists in monitoring injection rate and network load and in keeping the frequency low enough so that the NoC is forced to operate around its saturation point [10]. We see this technique as one possible implementation of a more general RMSD policy, in which the DVFS controller slows down the network as much as possible while guaranteeing that it can still sustain the injection rate. RMSD minimizes power at the cost of a highly increased network delay, as we will show in Sec. III.

In the context of a Chip Multi-Processor (CMP), Chen et al. explored DVFS techniques that aim at reducing the NoC power while targeting a given latency of the NoC and the last-level cache [11]. These techniques differ from RMSD as they target *delay* rather than *rate*. Our work has features in common with Chen et al.'s work, in that we also report a delay-based technique that we call Delay-based Max Slow Down (DMSD). However, we do not restrict our study to the CMP case, and justify through a detailed analysis and experimental results why a delay-based DVFS policy offers a better power-delay trade-off than a rate-based one.

III. Rate-based Max Slow Down (RMSD)

We assume that a *network clock* of frequency F_{noc} and a *node clock* of frequency F_{node} are available at the NoC and at each injection node, respectively. For simplicity, let F_{node} be fixed and equal for all the nodes[1]. Aim of the DVFS policy is to slow down the network clock with respect to the node clock (i.e., $F_{noc} < F_{node}$) to reduce the power consumption, without affecting the saturation throughput.

The tuning range of F_{noc} is $[F_{min}, F_{max}]$ and is set by the voltage-controlled oscillator of a PLL inside the DVFS controller. Without loss of generality, let the fixed value of F_{node} be equal to the maximum value F_{max}.

Suppose that nodes inject flits at rate λ_{node}, expressed in flits per node clock cycle. Equivalently, the injection rate in flits per second seen by each node is $R_{node} = \lambda_{node} F_{node}$, and seen by the NoC can be defined similarly as $R_{noc} = \lambda_{noc} F_{noc}$. To evaluate the corresponding λ_{noc}, i.e. the injection rate seen by the NoC and measured in flits per network clock cycle, we can set $R_{node} = R_{noc}$ and obtain:

$$\lambda_{noc} = \lambda_{node} \frac{F_{node}}{F_{noc}}. \quad (1)$$

Thus, when DVFS is applied and the network clock is slower than the node clock, the network sees more flits injected per network clock cycle and operates closer to its saturation point.

The main idea of RMSD is to minimize the network power consumption by slowing down F_{noc} as much as possible while

[1]We prefer to keep the exposition simpler and more intuitive, but a more general treatment with different and variable node frequencies is possible.

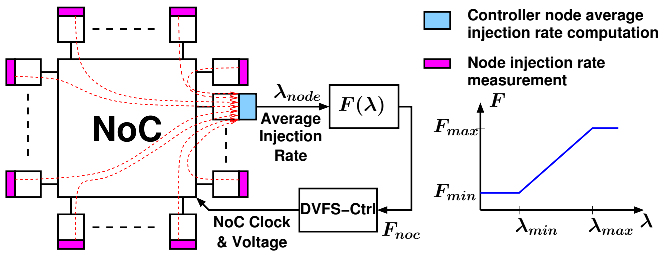


Fig. 1. Global controller in RMSD receives injection rate from NoC nodes and computes the NoC clock frequency.

preserving the maximum throughput. This is obtained by having the network work at an injection rate near but still below its saturation point. We define this target rate as λ_{max}. By setting $\lambda_{noc} = \lambda_{max}$ in (1), we obtain:

$$F_{noc} = F_{node} \frac{\lambda_{node}}{\lambda_{max}}. \quad (2)$$

For a given value of λ_{node}, by choosing F_{noc} as in (2) the network injection rate will be constant and equal to λ_{max}. As a result, the average network latency will also be constant.

The frequency-scaling law in (2) is valid whenever F_{noc} is in range $[F_{min}, F_{max}]$. This frequency range corresponds, through (2), to a range of node injection rates. The network injection rate is equal to λ_{max}, and the latency is constant, as long as the node injection rate stays within this range.

Under the assumption that $F_{node} = F_{max}$, we can easily determine this range of node injection rates. In particular, from (2) we obtain $F_{noc} = F_{max}$ when $\lambda_{node} = \lambda_{max}$, and $F_{noc} = F_{min}$ when $\lambda_{node} = \lambda_{max} F_{min}/F_{max}$. We define this lower node injection rate as λ_{min}. When λ_{node} is outside the range $[\lambda_{min}, \lambda_{max}]$, the network clock frequency is clipped to either F_{min} or F_{max}, as graphically shown in Fig. 1. The figure also depicts a possible RMSD architecture. The transmitting nodes periodically measure the injection rate as number of injected flits divided by the number of elapsed network clock cycles. This information is sent to one specific controller node (e.g. a central node in a mesh). The average injection is computed and used by the DVFS controller to set the network clock frequency and the corresponding voltage.

We implemented the RMSD policy in a micro-architectural cycle-accurate NoC simulator based on Stanford's Booksim [7]. We modified Booksim in various ways, but primarily by decoupling the network frequency from the nodes frequency.

Fig. 2 reports simulation results obtained in case of uniform traffic in a 5×5 mesh with dimension-ordered routing, routers with 8 virtual channels, 4 buffers per channel, 20 flits per packet. We set λ_{max} 10% lower than the saturation rate, which is 0.42 in this case. F_{node} is set equal to 1 GHz and F_{noc} varies between $F_{min} = 333$ MHz and $F_{max} = 1$ GHz.

Fig. 2(a) shows how the latency curve is modified by the RMSD strategy. As expected, the average *latency expressed in network clock cycles* is constant as long as the node injection rate is within the range $[\lambda_{min}, \lambda_{max}]$. Let us define the latency as L_{noc}. Since the network clock has been slowed down with

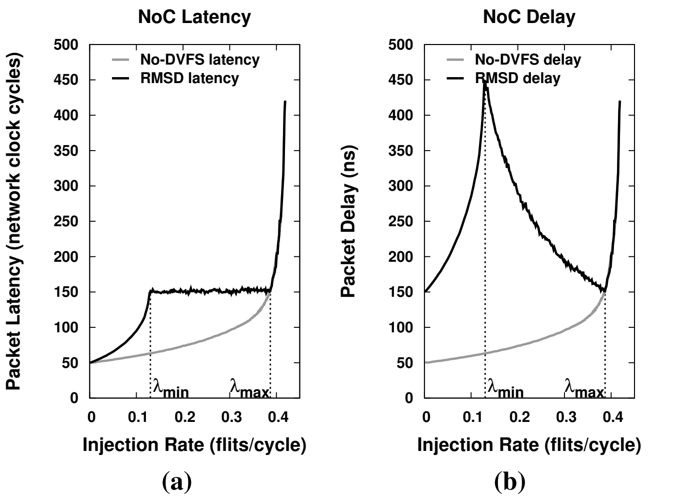


Fig. 2. RMSD vs No-DVFS. Network latency (a) and delay (b) for a 5×5 mesh with dimension-ordered routing, routers with 8 virtual channels, 4 buffers per channel, F_{node}=1 GHz, F_{min}=333 MHz, F_{max}=1 GHz.

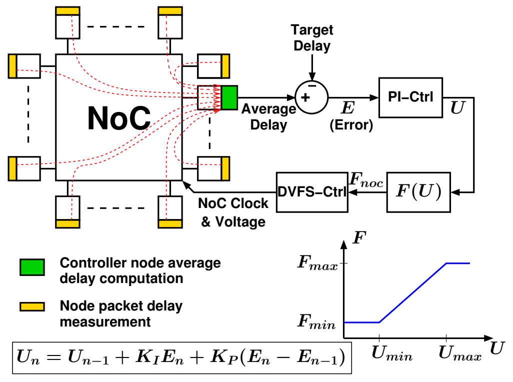


Fig. 3. PI loop controller in DMSD receives packet delay from NoC nodes and automatically sets the NoC clock frequency.

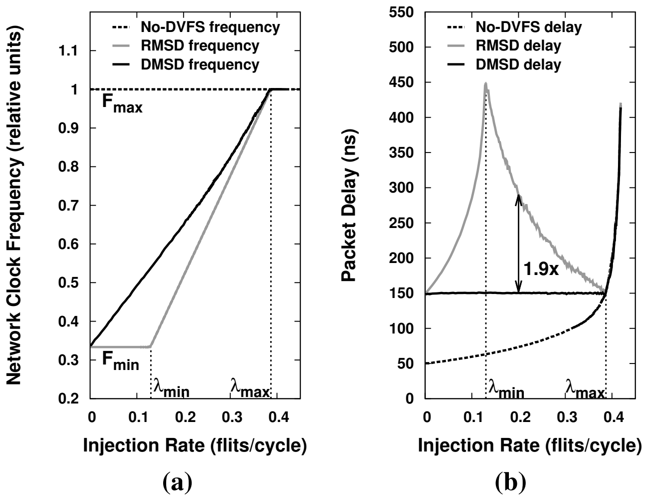


Fig. 4. DMSD, RMSD, and No-DVFS. Clock frequency (a) and delay (b), magenta under the same scenario settings as Fig. 2.

respect to the node clock, it is interesting to see the actual *delay in seconds*, which is the ratio L_{noc}/F_{noc}. Fig. 2(b) reports the delay curve. In the range $[\lambda_{min}, \lambda_{max}]$ the delay decreases because L_{noc} is constant and F_{noc} increases. In $[0, \lambda_{min})$, the delay increases because F_{noc} is fixed to F_{min} and L_{noc} increases due to the increasing load. As a result, the delay curve is non-monotonic with very long delays around the peak, about 9× greater than the original delay.

This non-monotonic behavior was observed for the first time in a context of DVFS policies for controlling the power of queue-based systems with a single server model [12], but was never observed before in the context of an NoC with DVFS.

In summary, the RMSD policy minimizes power consumption at the price of a large *delay* penalty. Notice that *throughput* is not affected, as long as the network does not saturate. RMSD is therefore useful only for applications that are not sensitive to delay. When delay matters, for instance in request-reply traffic, RMSD would be an inefficient choice.

IV. Delay-based Max Slow Down

In a DMSD architecture a specific node is dedicated to receiving measurements and implementing the DVFS policy, like in an RMSD architecture. DMSD needs, however, periodic measurements of end-to-end packet delays rather than of injection rate. Therefore, the *receiving* nodes, and not the transmitting ones, measure the delay. Measuring packet delay requires adding a timestamp on each packet, whose overhead is almost negligible if the delay is encoded in unused bits of the header flit [2]. The time counter and the circuitry for measuring the delay in each node consume only few gates.

The controller node computes an average delay and subtracts from it a *target delay*. This difference is the error signal of a closed-loop controller that tunes the network frequency in such a way to minimize the error. Fig. 3 shows a possible architecture with a proportional-integral (PI) controller. It is well known that by properly setting the PI controller gains, K_I and K_P, stability is guaranteed.

To simulate an NoC with the DMSD policy, and to compare it with the RMSD one, we used the same modified Booksim simulator. We experimented with different values of PI parameters and found a good compromise between stability and reactivity with K_I=0.025 and K_P=0.0125. These values have been used to obtain all the results reported in this paper. The control update period, which is also the interval period for delay transmission, does not need to be short [2]. We verified through simulations that 10,000 clock cycles (at the highest frequency) are sufficient. Therefore, not only the traffic overhead for sending the measurement information to the controller is negligible, but also the time for receiving and processing the measurements, and for control actuation, is a negligible fraction of the control update period. This makes the controller scalable to large networks (e.g. 8×8).

Fig. 4(a) compares the network clock frequency set by DMSD and RMSD controls, under the same conditions of Fig. 2. As expected, RMSD is more aggressive in tuning

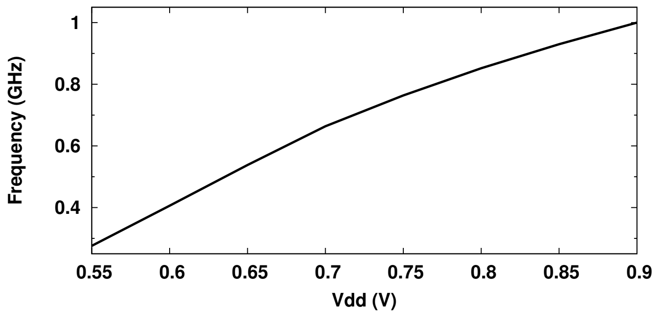


Fig. 5. Network clock frequency vs Vdd voltage in 28 nm FDSOI technology.

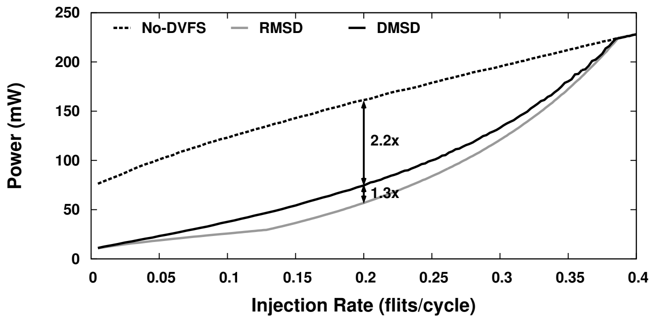


Fig. 6. DMSD vs RMSD and No-DVFS: Network power.

frequency, which is always less than or equal to the DMSD case. We expect then less power consumption in RMSD. Fig. 4(b) reports the delay in the DMSD case, with target set to 150 ns, which is the value of RMSD at injection rate λ_{max}. The figure shows that the PI controller manages to keep the RMSD delay constant in the entire range up to λ_{max}.

Fig. 4(b) shows that the RMSD delay is significantly greater than the DMSD delay. It is then interesting to evaluate if the power advantage of RMSD over DMSD is more or less than the delay advantage of DMSD over RMSD.

A. Power consumption evaluation

To evaluate the power consumption we synthesized with Synopsys Design Compiler an RTL version of the router used in simulations, targeting a low-power standard-cell library in a 28-nm FDSOI CMOS technology. To obtain the relationship between scaled voltage and frequency, we first extracted the Spice netlist of the router after synthesis. Then we simulated it with Mentor Graphics' Eldo transistor-level simulator with an input pattern that activates the critical path. The result of this analysis is shown in Fig. 5, which reports the maximum clock frequency of the router at a given Vdd voltage. We set the range of frequency for the following power evaluations from $F_{min} = 333$ MHz to $F_{max} = 1$ GHz, which results to voltage range from 0.56 V to 0.9 V.

To estimate power accurately, we took advantage of the fact that Booksim is a cycle accurate simulator and permits to save information of activity in the links, buffers, crossbar, etc. Therefore we imported the simulated activity information in Synopsys power estimation tool and obtained an accurate power estimation for any input rate, any router in the NoC, and any frequency-voltage pair chosen by the DVFS controller[2].

Fig. 6 compares the total NoC power, obtained summing the power of all routers and links, as a function of the injection rate. The conditions are the same of delay curves in Figs. 2 and 4. As expected, both RMSD and DMSD consume significantly less power than the No-DVFS baseline, with RMSD being more power efficient than DMSD. When we analyze the power-delay trade-off, however, we realize that DMSD is more efficient: at 0.2 injection rate, for instance, Fig. 6 shows that indeed DMSD consumes 30% more power than RMSD, but it also has a 90% lower delay, as clear from Fig. 4(b).

[2]Even though we let the controller choose any frequency in its range (and so also any voltage), the results remain valid in case of discrete values.

V. Experiments with Synthetic Traffic

We evaluated delay and power under various synthetic traffic conditions. Figs. 7(a)-(d) report the delay curves for four traffic patterns: tornado, bit-complement, transpose, and neighbor. Figs. 7(e)-(h) report the curves of power vs injection rate. The results are consistent with our previous observations. Independently from the traffic pattern, both RMSD and DMSD can save a significant amount of power compared to the No-DVFS case ($2.2\times$ and more at, for instance, 0.2 injection rate). DMSD consumes more power than DMSD (20%-40%), but this is more than compensated by the delay reduction of RMSD over DMSD (more than $2\times$ at 0.2 injection).

For the case of uniform traffic pattern, we performed a sensitivity analysis of our results when some of the router and NoC parameters are varied. In particular we changed the number of virtual channels (2, 4, and 8), the number of flit buffers per virtual channel (4, 8, and 16), the size in flits of the NoC packet (10, 15, and 20), and the size of the mesh (4×4, 5×5, and 8×8). The results in Fig. 8 confirm that the delay-power trade-off tips in favor of DMSD over RMSD under any of the considered variations.

VI. Experiments with Multimedia Traffic

To evaluate the power-delay trade-off in a realistic scenario, we selected two relevant applications from the multimedia domain, an MPEG4 Encoder (H.264) and a Video Conference Encoder (VCE) [13].

The two directed graphs in Fig. 9 are hybrid representations of the two applications, with both communication and mapping information. The mapping is represented by the position of the computation blocks (graph vertices) in a 4×4 mesh for H.264 and in a 5×5 mesh for VCE. The communication is represented by the graph edges and their weights, which correspond to the amount of packets exchanged for encoding a single frame of a multimedia stream.

To evaluate the performance of the two benchmarks we further modified Booksim in order to support custom traffic matrices, with and without DVFS. We simulated different frame rates and collected the activities inside the routers and in the links necessary for an accurate power estimation.

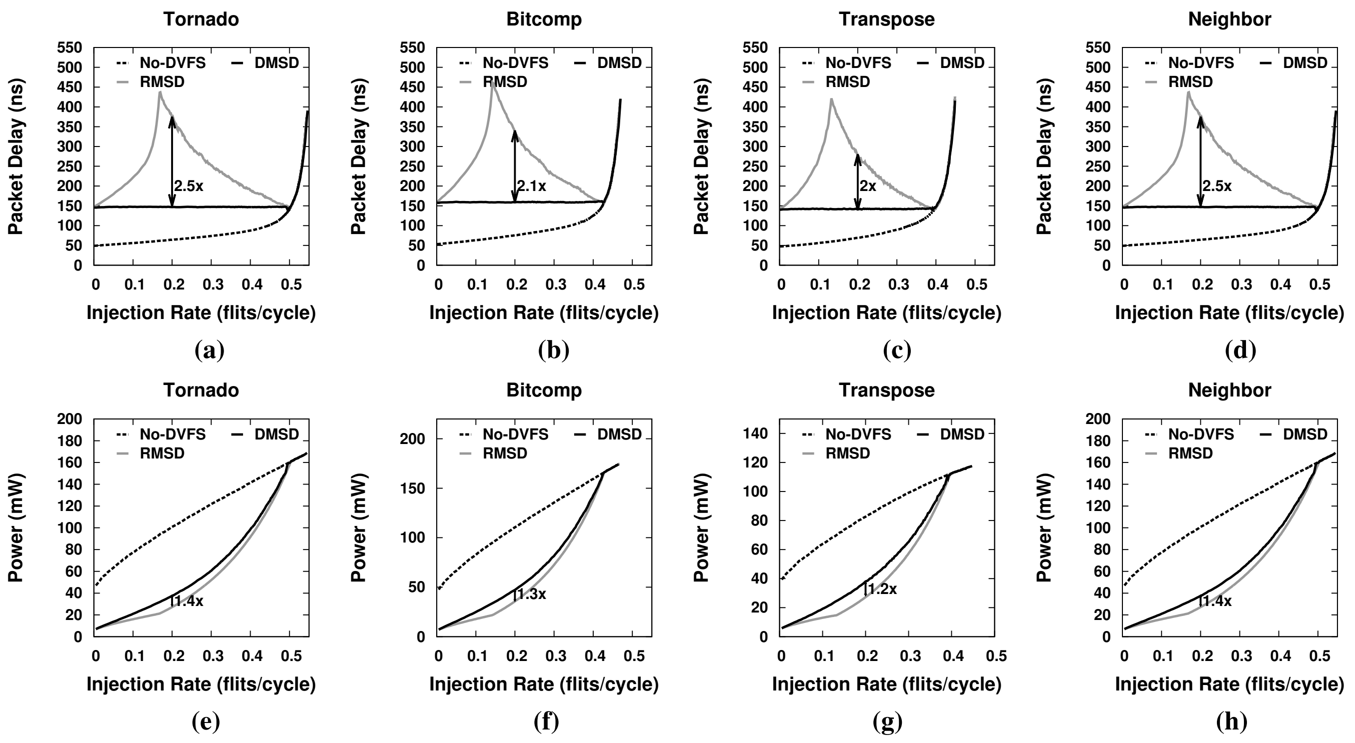


Fig. 7. RMSD vs DMSD delays and power under synthetic traffic scenarios: Tornado (a)(e), Bit-complement (b)(f), Transpose (c)(g) and Neighbor (d)(h).

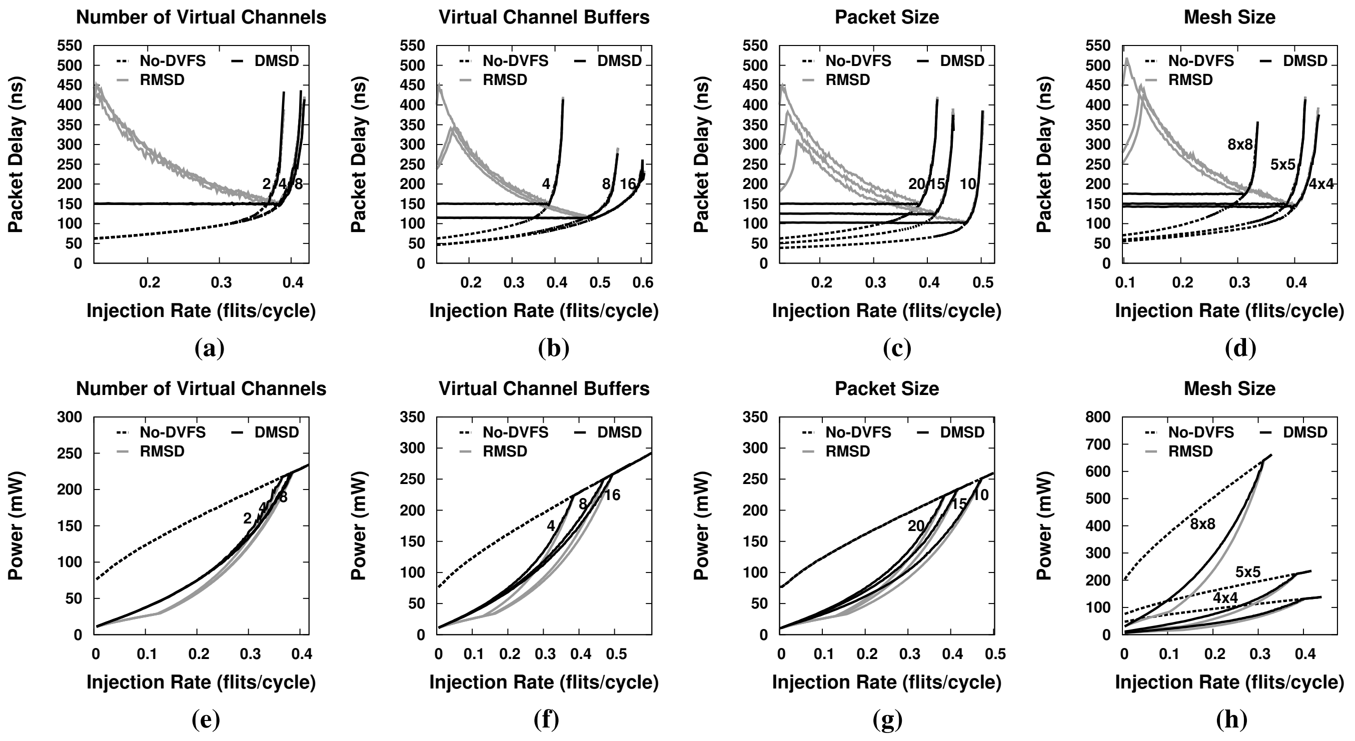
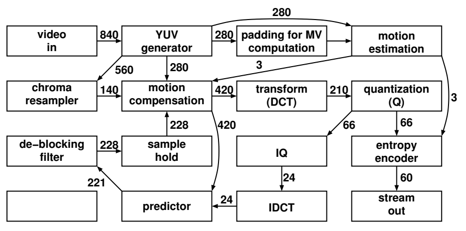
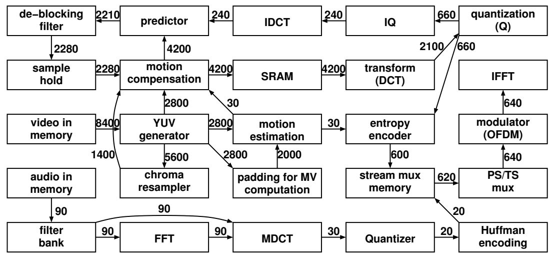


Fig. 8. RMSD vs DMSD delays and power, sensitivity analysis under uniform traffic, when varying: virtual channels (a)(e), buffer size (b)(f), packet size (c)(g), mesh size (d)(h).

The results of our analysis are reported in Fig. 10. The injection rate in x-axis is proportional to the *application speed*, which is normalized to the case of 75 frames/second. Once again, experimental results confirm that even in the realistic

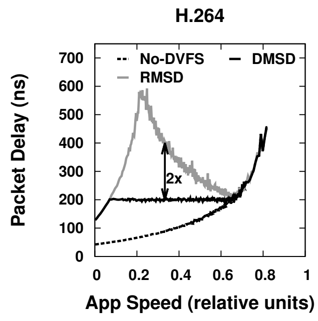


(a)

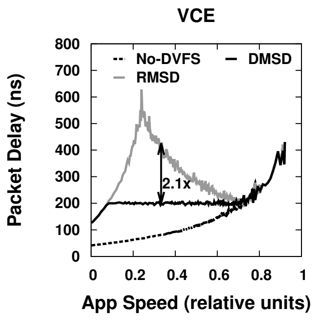


(b)

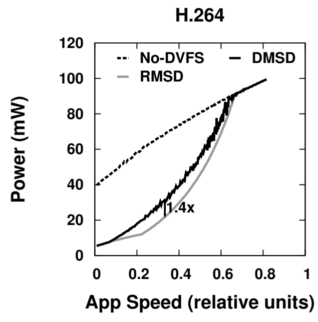
Fig. 9. H.264 (a) and VCE (b) communication graphs and NoC mapping.



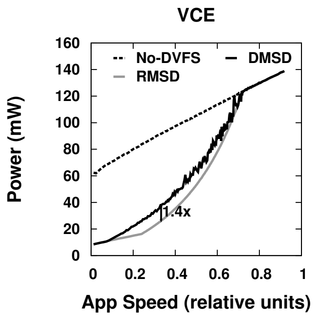
(a)



(b)



(c)



(d)

Fig. 10. RMSD vs DMSD delays and power under multimedia traffic. Delay: H264 (a), VCE (b). Power: H264 (c), VCE (d).

multimedia scenario, the additional power saving of an RMSD policy corresponds to a significant increase of the average NoC delay. The performance of encoders like those considered here, is often expressed in terms of application latency (in milliseconds). It is clear that the additional delay caused by the DVFS policy can significantly compromise the performance. For this reason we believe that minimizing the NoC power consumption at the cost of a largely increased application delay, as it happens with an RMSD policy, is not the best solution. A better trade-off will be obtained, instead, if the target delay is kept constant, and the power is minimized under the delay constraint, as it happens with a DMSD policy.

VII. Conclusion

In this paper we analyzed the power-delay trade-off in NoCs with DVFS. We showed that an aggressive strategy that minimizes power consumption disregarding NoC delay pays an excessive penalty in delay performance. We propose instead to set a target delay and to minimize power under that constraint. We implemented a proportional-integral control-loop that measures the average delay and makes sure that the target is not exceeded. We found that this strategy provides a better power-delay trade-off under various traffic scenarios. A sensitivity analysis shows that this conclusion is true under variations in the number of virtual channels, buffers, packet size, and mesh size.

References

[1] P. Salihundam *et al.*, "A 2 Tb/s 6x4 mesh network for a single-chip cloud computer with DVFS in 45 nm CMOS," *IEEE J. Solid-State Circuits*, vol. 46, no. 4, pp. 757–766, Apr. 2011.

[2] X. Chen *et al.*, "In-network monitoring and control policy for DVFS of CMP networks-on-chip and last level caches," *ACM Trans. on Design Automation of Electronic Systems*, vol. 18, no. 4, pp. 1–21, Oct. 2013.

[3] A. K. Mishra *et al.*, "A case for dynamic frequency tuning in on-chip networks," in *Proc. 42nd Int. Symp. Microarchitecture (MICRO-42)*, Dec. 2009, pp. 292–303.

[4] L. Guang, E. Nigussie, L. Koskinen, and H. Tenhunen, "Autonomous DVFS on supply islands for energy-constrained NoC communication," in *Arch. Comput. Sys. ARCS 2009*, ser. Lect. Notes Comput. Sc., 2009, vol. 5455, pp. 183–194.

[5] M. K. Yadav, M. R. Casu, and M. Zamboni, "LAURA-NoC: Local automatic rate adjustment in network-on-chips with a simple DVFS," *IEEE Trans. Circuits Syst. II*, vol. 60, no. 10, pp. 647–651, Oct. 2013.

[6] L. Shang, L.-S. Peh, and N. K. Jha, "Dynamic voltage scaling with links for power optimization of interconnection networks," in *Proc. 9th Int. Symp. High-Performance Computer Architecture (HPCA)*, 2003, pp. 123–124.

[7] N. Jiang *et al.*, "A detailed and flexible cycle-accurate network-on-chip simulator," in *Proc. Int. Symp. Performance Analysis Systems and Software (ISPASS)*, 2013, pp. 86–96.

[8] Q. Wu, P. Juang, M. Martonosi, L.-S. Peh, and D. W. Clark, "Formal control techniques for power-performance management," *IEEE Micro*, vol. 25, no. 5, pp. 52–62, Sep.-Oct. 2005.

[9] U. Y. Ogras, R. Marculescu, and D. Marculescu, "Variation-adaptive feedback control for networks-on-chip with multiple clock domains," in *Proc. 45th Design Automation Conference (DAC)*. ACM Press, 2008, pp. 614–619.

[10] A. Jantsch and L. Guang, "Adaptive power management for the on-chip communication network," in *Proc. 9th EUROMICRO Conf. on Digital System Design (DSD'06)*. IEEE, 2006, pp. 649–656.

[11] X. Chen *et al.*, "Dynamic voltage and frequency scaling for shared resources in multicore processor designs," in *Proc. 50th Design Automation Conference (DAC)*. ACM Press, 2013, pp. 114:1–114:7.

[12] A. Bianco, M. R. Casu, P. Giaccone, and M. Ricca, "Joint delay and power control in single-server queueing systems," in *Proc. IEEE Online Conf. on Green Communications (GreenCom)*, Oct. 2013, pp. 50–55.

[13] K. Latif, "Design space exploration for MPSoC architectures," Ph.D. dissertation, Univ. of Turku, Turku Center for Computer Science (TUCS), Finland, Dec. 2013. [Online]. Available: http://www.doria.fi/handle/10024/93883