

# Efficient DVFS Operation in NoCs Through a Proper Congestion Management Strategy

José V. Escamilla<sup>1</sup>(✉), José Flich<sup>1</sup>, and Pedro Javier García<sup>2</sup>

<sup>1</sup> Universitat Politècnica de València, Valencia, Spain  
joseslo@gap.upv.es, jflich@disca.upv.es

<sup>2</sup> Universidad de Castilla-La Mancha, Albacete, Spain  
pedrojavier.garcia@uclm.es

**Abstract.** As technology advances, applications demand more and more computing power. However, achieving the required performance is not nowadays the single target, as reducing power consumption has become a key issue. In that sense, power-control mechanisms such as Dynamic Voltage and Frequency Scaling (DVFS) are introduced in order to dynamically adapt frequency and voltage to the actual computing-power demands of applications. However, these techniques may not be as efficient as expected due to delays caused by frequency-voltage changes. Furthermore, data flows generated at high rates may cross slow voltage-frequency islands, thereby leading to congestion inside the on-chip network. To alleviate this problem, we propose a combined DVFS and congestion management strategy. Specifically, the policies to adjust DVFS levels are tuned cooperatively with the congestion management strategy, leading to power-saving achievements of up to 26 % and latency improvements for non-congested traffic of up to 43 %.

## 1 Introduction

Nowadays, High-Performance Computing (HPC) and multimedia-oriented applications and services demand increasing computing power. In order to satisfy this demand, manufacturers take advantage of the advances in integration-scale technology to include as many computing resources as possible into the same die. This trend has led to advanced designs in manycore chips. Regardless the specific design, these platforms require an on-chip network (NoC) [1] to support communication among all the processing and/or storage nodes. The NoC must provide high bandwidth and low latency, otherwise processing nodes will slow down as they have to wait for long to receive necessary data. Hence, the design of the NoC presents unavoidable challenges.

Among these challenges, a still open issue is how to efficiently deal with congestion situations, i.e. scenarios where any number of network paths are clogged, mainly due to oversubscribed ports (hotspots). Indeed, congestion may lead to a severe degradation in network performance if no countermeasures are taken. Another challenge is reducing power consumption. Technology has reached power and thermal limits, thus limiting clock frequency and voltage. Moreover, in

battery-powered devices, energy must be efficiently managed so as to maximize working time of the device. For these reasons, the current trend is to provide manycore chips with power-control mechanisms.

One of the most popular mechanisms is Dynamic Voltage and Frequency Scaling (DVFS) [12]. Basically, it consists in adapting the frequency and voltage based on the actual computing-power demand. Reducing the frequency and voltage leads to a significant reduction in power consumption, thus saving unneeded energy. However, DVFS must be carefully designed since reducing the working voltage and frequency may reduce also network performance. Thus, finding out the optimal conditions to increase or decrease the working voltage and frequency is critical to achieve the best trade-off between power saving and network performance. Additionally, voltage and frequency changes are usually performed in steps or levels, and those changes cause severe power and delay penalties, thus, demanding for a proper policy.

A DVFS change causes inherently the system to halt for a small period of time (due to the electronic limits) [15]. Recent proposals shadow this effect by setting different DVFS regions, called Voltage and Frequency Islands (VFIs) [11, 14]. In this way, frequency and voltage for a given island become independent as they are only driven by the metrics obtained from such island. From the efficiency point of view, VFI islands achieve an undeniable enhancement [11] as applications running concurrently may have different needs, some maximizing performance while others minimizing power. Nonetheless, VFIs still pose new challenges. Data flows may cross several VFIs working at different levels. Thus, the crossing from a high-frequency VFI to a low-frequency VFI will slow down the flow, potentially leading to a congestion situation appearing on VFI boundary. Moreover, congestion may be propagated throughout the high-frequency VFI network.

Summing up, DVFS-based systems need a proper policy to perform frequency-voltage transitions and, on the other hand, need to avoid congestion when VFIs are used. To address both issues, we propose adapting a congestion-control mechanism called ICARO [4] (Internal-Congestion-Aware Hol-blocking RemOval) to DVFS-based systems with VFIs. By doing this, performance is maintained despite of the DVFS-transition delays and congestion is alleviated despite of data flows crossing VFIs with different levels. ICARO congestion metrics will be used to implement the DVFS policy to perform voltage-frequency changes. We target different possibilities to plug a congestion control mechanism with a DVFS policy. Several solutions are presented which improve different key metrics, such as power consumption or message latency. Results show that we achieve improvements on network latency of 43 % for non-congested traffic with a power overhead of approximately 8 %. For the second solution a gain of 26 % on power consumption, with an improvement on latency of 2 % at the cost of losing throughput and, finally, the last proposal achieves an improvement on latency of 19 % with a power saving gain of 20 %. As networks-on-chip consume up to the 36 % of the total chip power [17, 18], the benefits of our proposal may improve substantially the overall power consumption.

The paper is organized as follows. Section 2 shows related work in DVFS and congestion-control mechanisms. Section 3 describes the ICARO-DVFS method. Section 3.6 shows analysis results for different DVFS scenarios combined with ICARO. Finally, Sect. 4 shows conclusions and future work.

## 2 Related Work

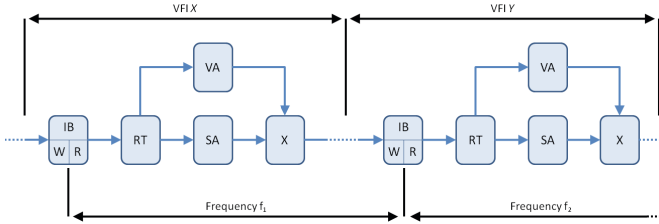
Related to DVFS, one key issue is the voltage-frequency regulator (VR) due to the high delays caused when changing the voltage-frequency level. DVFS regulators are designed either off-chip or on-chip. Off-chip regulators support high amounts of power, but they are slow. By contrast, on-chip regulators are very fast but expensive in terms of area and do not support much power. In [19], a hybrid scheme using both types of regulators is proposed. In systems using VFIs, the more VFIs are implemented, the more power efficiency is achieved, hence having one VFI per node would be the best case. Under this premise, in [9] authors propose a per-core VFI approach based on on-chip VRs. Despite authors state that area overhead would not be an obstacle to implement their proposal, in a newer study [19] they discard this approach due to the large area required to implement so many on-chip VRs, supporting their arguments on results obtained in [8]. An accurate DVFS model is described [15] for different real architectures, comparing with values from real systems. We use values of voltage/frequency levels (Table 2) and level change delays (Table 5) to model DVFS in our simulator.

Related to congestion, we find several solutions. Solutions for buffered NoCs are based on monitoring buffer occupancy and collecting congestion information from neighboring nodes. A congestion-free path is then used to avoid hotspot areas. In this way, RCA [5] uses multiple global metrics collected from the whole network to select at each router the output port which messages are forwarded through. However, a vicious cycle may be created since the information used to avoid the congested areas is included in the congested messages. Besides, adapting the routes to avoid hotspots may result in moving the location of such hotspots from one place to another. Moreover, avoiding hotspots may be impossible if all the flows have the same target (e.g. the memory controller).

Another solution based on adaptive routing is PARS [2], that uses a dedicated subnetwork to send congestion metrics about the buffer state at certain routers, then using these metrics to select paths that avoid hotspots. However, the problems regarding unavoidable hotspots or “hotspot reallocation” may still appear. Similarly, in [10] a token-based flow-control mechanism is proposed which uses dedicated wires to send router status information (token) used to make routing decisions and bypass router pipeline stages. However, this proposal is focused on reducing network latency but not by facing congestion harmful effects. In [16] authors collect congestion information from the whole network to make routing decisions. However, in this proposal the congestion information is collected by piggybacking the links status into the packets header. In the next section we describe ICARO [4] which attacks congestion in a more efficient manner.

**Table 1.** DVFS levels assumed in the ICARO-DVFS mechanism (obtained from [15])

DVFS level	Voltage(V)	Freq(GHz)	DVFS level	Voltage(V)	Freq(GHz)
Level 1	1.30	3.074	Level 4	1.15	2.281
Level 2	1.25	2.852	Level 5	1.10	1.932
Level 3	1.20	2.588	Level 6	1.05	1.540

**Fig. 1.** Two consecutive routers belonging to different VFIs (at the boundary delimiting such VFIs).

### 3 ICARO-DVFS Implementation

#### 3.1 Dynamic Voltage and Frequency Scaling

Our DVFS implementation changes voltage and frequency in levels, depending on the performance demand. Each level corresponds to a given pair of voltage-frequency fixed values (Table 1). Off-chip VRs are assumed.

DVFS levels are changed by monitoring the occupancy at router buffers. Every *poll\_period* cycles across all the monitored routers sample and report their occupancy level. If any of the buffer exceeds a  $Q_h$  threshold the DVFS level is decremented (voltage and frequency are incremented). Accordingly, if all the buffers exhibit an occupancy below the  $Q_l$  threshold ( $Q_h > Q_l$ ), the level is incremented (frequency and voltage are decremented).

#### 3.2 Voltage and Frequency Islands

ICARO-DVFS supports VFIs, so voltage and frequency changes can be applied per VFI domain. Each VFI has its own VR which collects metrics from the routers in its domain and changes voltage-frequency accordingly. However, routers at the boundaries between VFIs must be carefully designed as data flows will cross different domains. To address this issue, we implement mixed-clock/mixed-voltage buffers [3, 13] which enables to write and read at different frequencies.

Figure 1 shows the router implemented at a VFI boundary (see Sect. 3.6 for more information about routers architecture design). The *input buffer* stage is divided into two *sub-stages*, the write part belongs to the frequency domain of the upstream router while the read stage belongs to the current router

frequency domain. In this way, the rest of pipeline stages (after IB) are able to work normally at its corresponding frequency. The same is performed for the flow control logic. As credit-based flow control is used, the credits buffer at the downstream router works at both frequencies, at the upstream router frequency for writing and at the current router frequency for reading.

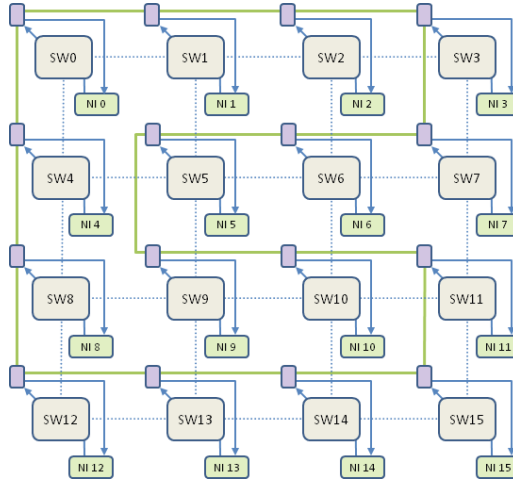
### 3.3 ICARO

Our proposal merges DVFS with ICARO [4], which tackles the congestion problem in a different way as the usual one. Specifically, ICARO focuses on reducing the impact of the Head-of-Line (HoL) blocking [7] derived from congestion situations. This harmful effect happens when congested flows share buffers with non-congested flows, then the former slowing down the latter and so degrading network performance. In order to deal with HoL-blocking, ICARO separates congested flows from non-congested ones by means of the different VNs implemented as disjoint virtual channels in the network. Note that congestion situations (i.e. congested flows) are not removed but their negative impact is reduced or even eliminated. Basically, ICARO detects congestion at routers, notifies congestion to the end nodes, and they react by steering packets through different virtual networks depending whether they cross congestion spots or not.

Congestion is detected at routers. A dedicated module at each router analyzes buffers of each port and, based on the buffer utilization, computes pending requests from each input port to each output port in order to detect contention (more than one request for a given output port). If contention is detected and caused by oversubscribed input ports (and lasts for a given threshold), then the output port is declared congested. Only if one output port changes its congestion state, then a notification is triggered from the router.

A congestion notification consists of the state of all output ports of the router (one bit per output port). Notifications are sent from routers to end nodes through a dedicated network (Congestion Notification Network or CNN) implemented as a  $k$ -width segmented ring where  $k = \log_2(nodes) + router_{radix} + 1$ . Figure 2 shows a CNN implementation for a  $4 \times 4$  mesh NoC. The CNN is made of  $N$  registers, each one owned by a router and connected to the next register through a  $k$ -width link. At each clock cycle, all the registers forward their data to the next register along the ring. If a router needs to inject a congestion notification, it checks its register state and injects the notification once the register becomes free. Notifications travel along the entire ring and are removed when they reach the register where they were injected from. Notifications are delivered to all end nodes and at each node the notification is processed and stored in a notification cache as *congested point* (pairs of values made up of {congested\_router, congested\_port}). Registers in this network are implemented as mixed-clock/mixed-voltage buffers as well to cope with different frequency domains.

Congestion isolation is performed at end nodes. To do so, ICARO uses two VNs: one congested-VN and one regular-VN. All flows are always mapped first to regular-VNs, but a *post-processor* module checks at each cycle the flit at the head



**Fig. 2.** CNN network example. Links in green: CNN interconnecting all CNN registers.

of the regular-VN. If the flit is a header and its path travels through a *congested point*, it is considered a congested flow so, then the flit and the remaining ones of the same message are relocated in the congested-VN relocating automatically all flits from such queue until a tail flit is found). Messages injected through a given VN never change to a different VN, thus achieving the isolation property pursued by ICARO.

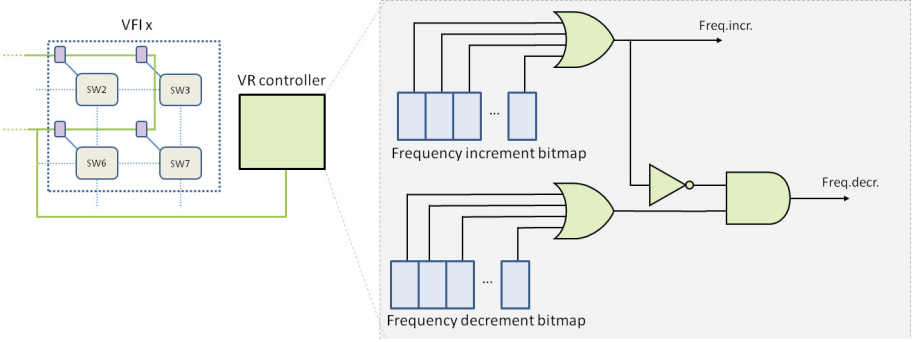
### 3.4 Merging ICARO with DVFS

Now we show how ICARO is coupled with DVFS. Basically, ICARO notifies congestion events through the CNN network. This network is extended to send DVFS notifications as well<sup>1</sup>. This is easily achieved by changing the ICARO controller implemented in the router. In addition to detect congestion events, the new logic monitors all input ports queue occupancy and sends two new events through the CNN. Figure 4 shows the new notification format. Two bits are added just indicating the router requests for a level increment or decrement.

Buffer occupancy is analyzed in each router and compared against  $Q_h$  and  $Q_l$  thresholds. If any of the queues exceeds the  $Q_h$  threshold the  $VFI_{inc}$  bit is set. Once all queues occupancy are below  $Q_l$  the  $VFI_{dec}$  bit is set. Only when any of those two bits change a DVFS notification is sent through the CNN network.

At each VFI domain, the VFI module reads the notification commands from the CNN network, keeping record of all routers  $VFI_{inc}$  and  $VFI_{dec}$  bits from its domain. Two  $n$ -length bit vectors (being  $n$  the number of routers in the

<sup>1</sup> In a typical DVFS implementation, a dedicated logic collects metrics from the network and deliver them to the logic that implements the VFI policy and drives its VR to carry this out. We take advantage of the CNN network to simplify this process.



**Fig. 3.** Voltage Regulator controller logic

```

if CNNreg.busy && idBelongsToVFI(CNNreg.routerID, thisVFI) then
  if CNNreg.FreqIncr then
    freqIncrVector[routerID] = 1;
    if |freqIncrVector then
      increaseFrequency(thisVFI);
    else if CNNreg.FreqDecr then
      freqDecrVector[routerID] = 1;
      if |freqDecrVector && ~|freqIncrVector then
        decreaseFrequency(thisVFI);
    else
      Do nothing
    end
  end
end

```

**Algorithm 1:** DVFS level change algorithm.

VFI domain) are implemented. VFI frequency/voltage is incremented when any of the routers request for such increment (even if a router is requesting for a decrement). Frequency decrement is performed when any router is requesting a decrement and none of the routers are requesting an increment. Figure 3 shows the logic, whereas Algorithm 1 shows the algorithm change DVFS levels.

### 3.5 Different ICARO-DVFS Alternatives

Besides the CNN extension, ICARO deals with different virtual networks (VNs) to decouple congested traffic from non-congested traffic. In the minimal implementation (2 VNs), just one VN is used to map congested traffic, the other one remains for non-congested traffic. As commented previously, to couple correctly ICARO and DVFS we need to sense occupancy of input ports queues. However, as we have differentiated VNs we have different options.

As a first alternative, we can sense all input ports queues, including both non-congested traffic VN and congested traffic VN. In this case, DVFS will raise

frequencies and voltages whenever traffic increases to levels where congestion appears in the VFI domain. This alternative is referenced to as *ICARO-2VN*.

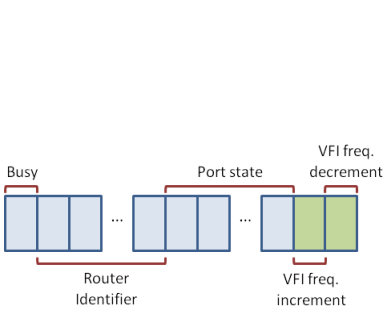
Another alternative is to raise frequencies and voltages only whenever the non-congested VN congests as well. In this case, the DVFS strategy will raise only when severe congestion appears in the network. In other words, when congestion caused by hotspot traffic affects background traffic in such a way in which causes regular-VNs to exceed  $Q_h$  threshold. It is supposed that this alternative will lead to more power-saving results. This alternative will be referenced to as *ICARO-1VN*.

Finally, a different alternative is to sense all input port's queue, but differently from *ICARO-2VN*, the DVFS strategy will be bounded to a more conservative frequency level. In this case, the maximum frequency will be set to  $\sim 2\text{GHz}$  (instead of  $\sim 3\text{GHz}$ ), corresponding to *Level 5* frequency on Table 1. The reasoning behind this strategy is the effect *ICARO* has on performance as will decouple congested traffic from non-congested one. Therefore, increasing frequency for performance reasons will become less critical. This alternative will be referenced to as *ICARO-2GHz*. In addition, this approach allows to simplify VRs by reducing the number of voltage-frequency levels provided. Notice that area consumed by VRs depends directly on the voltage-frequency levels provided.

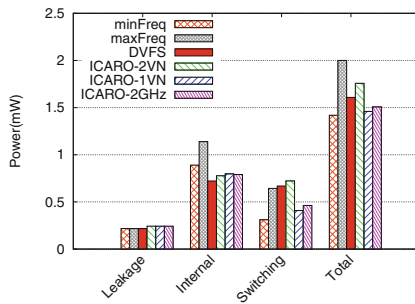
The three strategies will be analyzed in Sect. 3.6. All of them will be compared against three different strategies. The two first strategies will not use DVFS at all and will set the network both to minimum and maximum frequencies. They will be referenced to as *minFreq* and *maxFreq*, respectively and will allow us to set the low and up limits in terms of performance and power. The third strategy will be compounded of DVFS with the defined levels shown in Table 1 and sensing the VFI occupancy queues regardless of the congestion effects. This strategy will be referenced to as DVFS in the plots.

### 3.6 ICARO-DVFS Performance Analysis

**Simulation Environment.** We use gMemNoCsim, an in-house cycle-accurate event-driven NoC simulator. We model a 4-stage pipelined router: IB, RT, SA, X.



**Fig. 4.** CNN signal format in DVFS-based platforms



**Fig. 5.** Final power consumption.

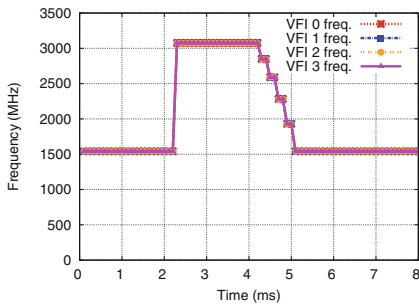
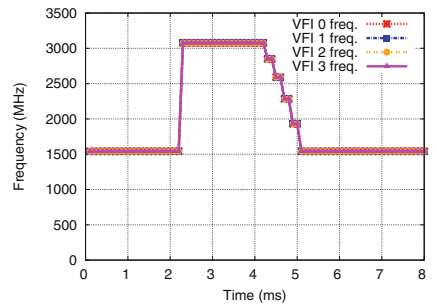
**Table 2.** Common simulation configuration.

Topology	8×8 2D regular mesh
Routing policy	XY
Switching technique	Wormhole (flit-level)
Flow control	credits
Flit size	128 bits
Message size	5 flits
Switch queue size	16 flits

At IB flits are stored into the buffer. In the case of routers at the VFI boundary, a *coupling IB stage* (see Sect. 3.2) is implemented with two substages: *IB\_W* (writing data) and *IB\_R* (read data). At RT, routing computation is performed, SA performs switch allocation, and at X stage the flit crosses the crossbar and leaves the router.

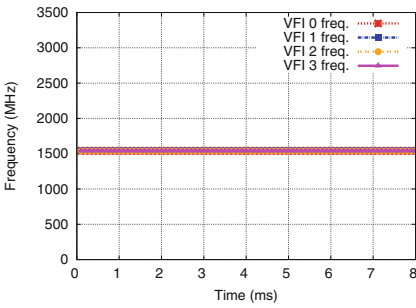
Regarding traffic patterns, as ICARO is a proposal intended to deal with irregular, bursty, hotspot-prone traffic patterns, we drew up a combined traffic pattern. This traffic pattern is composed of a light uniform background traffic (at a datarate of 0.01 flits/ns) and a hotspot component. Hotspots consist in several nodes receiving each one high data rates (3 flits/ns) from 4 nodes (4-to-1 hotspots). Hotspots are active only from time 20ms to 40ms. In this way, we have a background traffic which generates no congestion, plus an aggressive traffic which causes congestion, causing HoL-blocking to the background traffic. This compound traffic pattern emulates environments where light data flows (i.e.: cores running applications with light data demand) share the NoC resources with heavy traffic generated nodes running high data demand applications or hardware accelerators which tend to generate heavy data bursts, causing congestion as well.

Simulations with DVFS are performed using real voltage, frequency and delay values shown in Table 1. For power consumption measures we use Orion 3.0 [6].

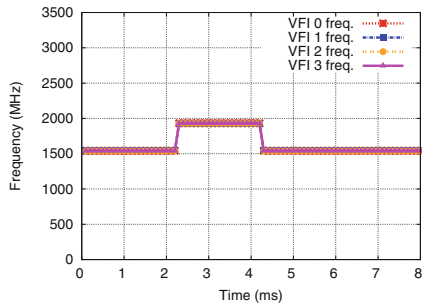
**Fig. 6.** VFIs frequencies for DVFS without ICARO.**Fig. 7.** VFIs frequencies for ICARO-2VN.

**Results.** In this section results for different configurations are shown. First, two configurations without DVFS are considered: *minFreq* for the chip running at the minimum frequency (1.540GHz) and *maxFreq* for the chip running at the maximum frequency (3.074GHz). Then, results for a DVFS scenario (without ICARO) are shown. Finally, results for the three ICARO-DVFS versions are shown. It is worth recalling that the main ICARO goal is to keep background traffic unaffected when dealing with congestion-prone traffic, such as the simulated hotspot traffic. For that reason results for background and hotspot traffic are shown separately.

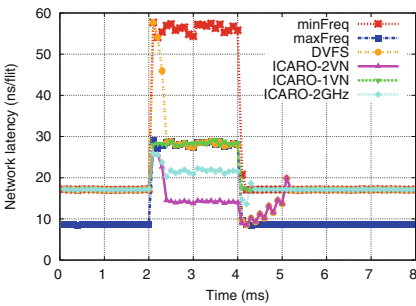
Figure 10 shows average network latency results. For the DVFS cases some peaks in latency can be seen when congestion background starts and ends. These peaks clearly show the overhead derived from the VR taking some time to perform the DVFS level changes. It is worth mentioning that the first peak is higher as the VR takes more time to change from the lowest frequency to the highest. After some analysis, we decided to increase from the minimum frequency to the maximum one since this involves only one transition, incurring much less penalty with respect to a step-by-step increase. As expected, the scenario with DVFS improves in power consumption the no-DVFS scenario, but at the cost of increasing network latency (Fig. 12) and decreasing throughput (Fig. 13).



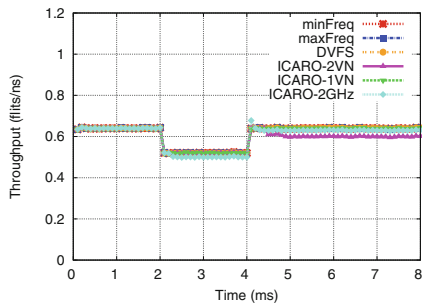
**Fig. 8.** VFIs frequencies for ICARO-1VN.



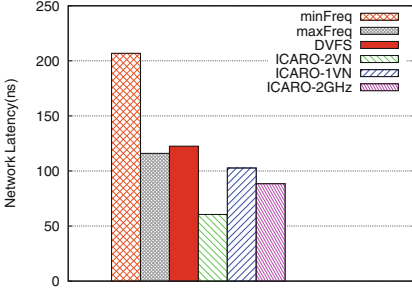
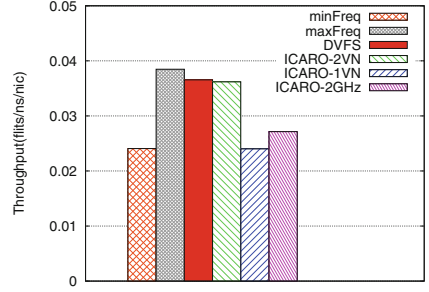
**Fig. 9.** VFIs frequencies for ICARO-2GHz.



**Fig. 10.** Network latency for background traffic.



**Fig. 11.** Throughput for background traffic.

**Fig. 12.** Final net. latency (all traffic).**Fig. 13.** Final throughput (all traffic).

However, according to Fig. 5, power consumption saving is more significant than performance degradation.

As can be seen in Fig. 10, ICARO proposals separate effectively background traffic from the hotspot one, preventing the HoL-blocking effect over the former. As expected, ICARO-2VN achieves the highest improvements in network latency (up to 43 %) since it takes into account all VNs to trigger the frequency-increment mechanism, and it is able to increment frequency to the maximum, but at the cost of increasing power consumption (8 %). Nevertheless, despite being the ICARO proposal with the highest power consumption, this case still keeps power consumption below “DVFS-alone” levels while improving latency.

Regarding the results for ICARO-1VN, note that only *regular-VNs* (VN 0) is taken into account to increase the VFI frequency and only background traffic is forwarded through this VN, so that this slight traffic flow is not enough to trigger the frequency-increment mechanism and all VFIs end up working at the minimum frequency. Despite running at the minimum frequency, we can see that the background network latency is kept in similar values to the DVFS case working at highest frequency. This is achieved by separating both traffic types, so preventing the HoL-blocking that the hotspot traffic could cause. In addition, as frequency is the lowest allowed, power saving is maximum, achieving an improvement of 26 %. However, as the system is working at the minimum frequency, throughput is lower than other cases that are running at higher frequencies. Nevertheless, background traffic achieves acceptable latency values.

Finally, ICARO-2GHz case could be considered as the best trade-off proposal. It takes into account all the available VNs but it is only allowed to increment frequency to the next step from the lowest frequency. This proposal does not achieve the best results in network latency and throughput, but nevertheless it improves the “DVFS-alone” case by 19 % with a significant power-saving improvement (20 %) with respect to DVFS, due to the lower frequency used.

## 4 Conclusions and Future Work

In this work a combination of DVFS scheme with a congestion management mechanism (ICARO) based on separating harmful traffic from non-harmful one

has been presented. Using the dedicated network used by ICARO in order to also deliver DVFS metrics for triggering frequency changes has been proposed. Three different approaches have been proposed in order to combine DVFS with ICARO. The first taking into account all VNs to trigger the frequency change, achieving the best latency improvements with a slight power consumption penalty. The second one only takes into account the *regular-VNs* (VN 0) to trigger frequency changes, achieving a power consumption improvement of 26 %. Finally, the third proposal limit the frequency increase until 2GHz (instead of 3GHz), with a latency improvement of 18 % and a power-saving gain of 20 %. As future work we plan to use messages latencies as congestion metric and implement support in-order delivery support.

**Acknowledgements.** This work was supported by the Spanish Ministerio de Economía y Competitividad (MINECO) and by FEDER funds under Grant TIN2012-38341-C04-01 and by Ayudas para Primeros Proyectos de Investigación from Universitat Politècnica de València under grant ref. 2370.

## References

1. Benini, L., De Micheli, G.: Networks on chips: a new SoC paradigm. *Computer* **35**(1), 70–78 (2002)
2. Chang, X., Ebrahimi, M., Daneshtalab, M., Westerlund, T., Plosila, J.: Pars an efficient congestion-aware routing method for networks-on-chip. In: 2012 16th CSI International Symposium on Computer Architecture and Digital Systems (CADS), pp. 166–171 (2012)
3. Chelcea, T., Nowick, S.: A low-latency FIFO for mixed-clock systems. In: IEEE Computer Society Workshop on VLSI, 2000. Proceedings, pp. 119–126 (2000)
4. Escamilla, J., Flich, J., Garcia, P.: ICARO: Congestion isolation in networks-on-chip. In: 8th International Symposium on Networks-on-Chip, 2014, NoCS 2014, pp. 159–166 September 2014
5. Gratz, P., Grot, B., Keckler, S.W.: Regional congestion awareness for load balance in networks-on-chip. In: HPCA, pp. 203–214. IEEE Computer Society (2008)
6. Kahng, A., Lin, B., Nath, S.: ORION3.0: a comprehensive NoC router estimation tool. *IEEE Embed. Syst. Lett.* **7**(2), 41–45 (2015)
7. Karol, M., Hluchyj, M., Morgan, S.: Input versus output queueing on a space-division packet switch. *IEEE Trans. Commun.* **35**(12), 1347–1356 (1987)
8. Kim, W., Brooks, D., Wei, G.Y.: A fully-integrated 3-level dc/dc converter for nanosecond-scale dvs with fast shunt regulation. In: 2011 IEEE International Solid-State Circuits Conference Digest of Technical Papers (ISSCC), pp. 268–270, February 2011
9. Kim, W., Gupta, M., Wei, G.Y., Brooks, D.: System level analysis of fast, per-core DVFS using on-chip switching regulators. In: IEEE 14th International Symposium on High Performance Computer Architecture, 2008, HPCA 2008, pp. 123–134, February 2008
10. Kumar, A., Peh, L.S., Jha, N.: Token flow control. In: 2008 41st IEEE/ACM International Symposium on Microarchitecture, 2008, MICRO-41, pp. 342–353, November 2008

11. Lackey, D., Zuchowski, P., Bednar, T., Stout, D., Gould, S., Cohn, J.: Managing power and performance for system-on-chip designs using voltage islands. In: IEEE/ACM International Conference on Computer Aided Design, 2002, ICCAD 2002, pp. 195–202, November 2002
12. Macken, P., Degrauwe, M., van Paemel, M., Oguey, H.: A voltage reduction technique for digital systems. In: 1990 IEEE International Solid-State Circuits Conference, 1990, 37th ISSCC. Digest of Technical Papers, pp. 238–239, February 1990
13. Marculescu, D., Choudhary, P.: Hardware based frequency/voltage control of voltage frequency island systems. In: Proceedings of the 4th International Conference on Hardware/Software Codesign and System Synthesis, 2006, CODES+ISSS 2006, pp. 34–39, October 2006
14. Ogras, U., Marculescu, R., Marculescu, D., Jung, E.G.: Design and management of voltage-frequency island partitioned networks-on-chip. *IEEE Trans. Very Large Scale Integr. Syst. (VLSI)* **17**(3), 330–341 (2009)
15. Park, S., Park, J., Shin, D., Wang, Y., Xie, Q., Pedram, M., Chang, N.: Accurate modeling of the delay and energy overhead of dynamic voltage and frequency scaling in modern microprocessors. *IEEE Trans. Comput.-Aided Des. Integr. Circ. Syst.* **32**(5), 695–708 (2013)
16. Ramakrishna, M., Gratz, P., Sprintson, A.: GCA: Global congestion awareness for load balance in networks-on-chip. In: 2013 Seventh IEEE/ACM International Symposium on Networks on Chip (NoCS), pp. 1–8, April 2013
17. Shang, L., Peh, L.S., Jha, N.: Dynamic voltage scaling with links for power optimization of interconnection networks. In: The Ninth International Symposium on High-Performance Computer Architecture, 2003, HPCA-9 2003, Proceedings, pp. 91–102, February 2003
18. Vangal, S., Howard, J., Ruhl, G., Dighe, S., Wilson, H., Tschanz, J., Finan, D., Singh, A., Jacob, T., Jain, S., Erraguntla, V., Roberts, C., Hoskote, Y., Borkar, N., Borkar, S.: An 80-tile sub-100-w teraFLOPS processor in 65-nm CMOS. *IEEE J. Solid-State Circuits* **43**(1), 29–41 (2008)
19. Yan, G., Li, Y., Han, Y., Li, X., Guo, M., Liang, X.: Agileregulator: a hybrid voltage regulator scheme redeeming dark silicon for power efficiency in a multicore architecture. In: 2012 IEEE 18th International Symposium on High Performance Computer Architecture (HPCA), pp. 1–12, February 2012