Sensor-wise methodology to face NBTI stress of NoC buffers

Davide Zoni and William Fornaciari
Politecnico di Milano – Dipartimento di Elettronica e Informazione
Via Ponzio 34/5, 20133 Milano, Italy
{zoni,fornacia}@elet.polimi.it

***Abstract*—Networks-on-Chip (NoCs) are a key component for the new many-core architectures, from the performance and reliability standpoints. Unfortunately, continuous scaling of CMOS technology poses severe concerns regarding failure mechanisms such as NBTI and stress-migration. Process variation makes harder the scenario, decreasing device lifetime and performance predictability during chip fabrication. This paper presents a novel cooperative sensor-wise methodology to reduce the NBTI degradation in the network on-chip (NoC) virtual channel (VC) buffers, considering process variation effects as well. The changes introduced to the reference NoC model exhibit an area overhead below 4%. Experimental validation is obtained using a cycle accurate simulator considering both real and synthetic traffic patterns. We compare our methodology to the best sensor-less round-robin approach used as reference model. The proposed sensor-wise strategy achieves up to 26.6% and 18.9% activity factor improvement over the reference policy on synthetic and real traffic patterns respectively. Moreover a net NBTI V_{th} saving up to 54.2% is shown against the baseline NoC that does not account for NBTI.**

I. Introduction

To cope with power/performance trade-off , the accepted solution to integrate multiple cores in a single chip poses new communication challenges, and Network-on-Chip (NoC) [1] emerges as a viable and effective design paradigm to manage performance and reliability requirements. However, the actual integration density is limited by the reliability of the device: rising of operating temperatures and physical failure mechanisms e.g. Negative Biased Temperature Instability (NBTI) or stress-migration, can seriously limit the device lifetime.

Considering CMOS circuits, NBTI occurs in PMOS transistors when $V_{gs} = -V_{dd}$, namely when the PMOS is active. Such *stress condition*, has been faced by increasing the threshold voltage (V_{th}) producing, as a side-effect, a degradation of the driving current. This can lead to performance loss or, in the worst scenario, to the stuck of the transistor. It has been shown that NBTI can increase $|V_{th}|$ by as much as $50\mathrm{mV}$ for devices operating at 1.2V or below [2], while the circuit performance degradation may reach 20% in 10 years [3]. On the other side, when a logic "1" is applied to the gate, i.e. $V_{gs} = 0$, NBTI stress is removed. This latter condition, called *recovery state*, induces a progressive, yet partial, recovery of the device threshold voltage. PMOS recovery can be also achieved by switching off the component, i.e. via power gating.

At the same time, *Process Variability* (PV), produces unexpected power/performance fluctuations, which are becoming a major fabrication challenge for the upcoming technology scales. In particular, it has been observed a latency degradation up to 40% and a leakage power variation on buffers of about 90% due to PV [4]. Although many works address the NBTI and PV considering either device, circuit or system levels, they are primarily focused on the computational logic of a multi-core chip. From the best of our knowledge, a comprehensive framework to manage NBTI issues in the on-chip networks considering the PV impact has not been proposed yet.

This paper provides a novel sensor-wise methodology to mitigate NBTI degradation in the on-chip network virtual channel (VC) buffers of multi-core architectures by reducing the stress period, also considering PV issues. It is worth noticing NBTI issues affect buffers on both combinatorial logic as well as on the storage logic [5], i.e. SRAM cells. The paper encompasses the following three aspects:

- *Network-on-Chip NBTI estimation framework* – Starting from HANDS [6], the proposed approach has been integrated in a multi-core simulator that can simulate NoC communication subsystems using both synthetic and real traffic scenarios. We provide an NBTI sensor library coupled with the simulation environment starting from a flexible analytical NBTI model from the available literature [7].
- *Reliability enhancement* – A sensor-wise NBTI mitigation methodology for router's buffers is proposed, providing significant NBTI improvement against sensor-less round-robin methodology, that is also developed as reference golden model. It is worth to notice that the proposed sensor-less round-robin policy represents the best approach we can cast for NBTI recovery purposes without any sensor-related information. Moreover, we provide a proof of the feasibility of the proposed approach through the area overhead estimation.
- *Cooperative approach* – Both sensor-wise and sensor-less approaches enhance the NBTI reduction thanks to the exploitation for each router of the information on incoming traffic from neighbor routers. The proposed strategy reduces the NBTI impact by reducing the buffers' stress periods. To this extent, our results show how the cooperative methodology can improve the recovery level against a simple non cooperative approach.

The paper is organized as follows. Section II reports an overview of the state-of-the-art on NBTI related methodologies, for both computational and communication logic. Section III describes the proposed sensor-wise and sensor-less NBTI mitigation strategies as well as the baseline NoC pipeline model employed. Experimental results are then reported and discussed in Section IV. Conclusions are drawn in Section V.

II. Related works

This section provides an overview of the state-of-the-art related to NBTI mitigation design methodologies considering also PV issues.

A linear programming based approach has been proposed in [8] to optimally drive the inputs to individual gates in order to prevent static NBTI fatigue. The work addresses the NBTI stress due to long stand-by periods. Li et al. exploits idle time in functional unit inside a processor core to recover NBTI degradation with a negligible performance loss and area overhead [9]. [10] proposed a multi-level technique to reduce the impact of NBTI degradation on the functional units of a high performance processor core. Li et al. observes how process variation can significantly influence on-chip network design choices [11]. To this extent, Ogras et al. studied the multiple voltage-island design technique applied to on-chip networks to face the PV issues [12].

Xin Fu et al. present a comprehensive approach to mitigate NBTI degradation considering NoC architectures [13]. This work discusses several approaches to effectively manage NBTI impact, focusing on different micro-architectural components, namely buffers and arbiters. However, [13] does not exploit the information on actual NoC traffic between each router pair, thus the solution looses the possibility to aggressively reduce NBTI degradation as well as to perform a NBTI/performance trade-off.

The tightly coupling between NoC performance and virtual channel buffer structure motivated the work iDEAL [14], a framework to design energy and area-efficient links that are capable of data transmission as well as data storage when required. The goal is a reduction in the router buffer size by controlling the repeaters along the links to adaptively work as link buffers during NoC congestion.

Last, [7] reviewed the main NBTI mitigation mechanisms providing also an accu- rate NBTI model that we adopted to extract absolute V_{th} NBTI values from our experimental data.

Despite the depth of each specific investigation, the lack of current literature resides mainly in the missing of a comprehensive methodology and the related supporting NoC analysis and design framework, capable to concurrently take into account several figures of merit such as NBTI and PV while exploring cost/performance/reliability trade-off in a way so flexible to enable considering alternative micro-architectural solutions for the on-chip network.

III. Proposed estimation flow

This section presents the proposed methodology through four different steps. First, Section III-A discusses two exploited architectural features to take advantage in our methodology and some NBTI background we exploited. Section III-B discusses the *round-robin sensor-less* approach (*rr_no_sensor*), that we use to assess the validity of our methodology. The sensor-wise methodology rationale and the logic modifications to the baseline NoC are detailed in Section III-C. Moreover, a discussion on the introduced area overhead is reported in Section III-D. The baseline router we consider is a 3-stage virtual channel pipelined router, with no packet mixing; it is implemented in the Garnet [15] available in GEM5 and it is not NBTI aware.

A. Methodology Background

NBTI has two phases *stress* and *recovery*. A buffer in a NBTI stress phase when it is storing at least one flit or when it is idle from the NoC point of view, namely there is an input configuration vector on its inputs even if it is meaningless. In our approach, a buffer is in the recovery phase if it is switched-off. Since the duty-cycle, or NBTI stress period, definition for NBTI purposes is quite different from the traditional duty-cycle definition, we defined the *NBTI-duty-cycle as*:

$$\text{NBTI-duty-cycle} := \frac{\text{stress-cycles}}{\text{stress-cycles} + \text{recovery-cycles}} * 100$$

where *stress-cycles* and *recovery-cycles* are the NBTI stress period and the NBTI recovery period respectively.

NBTI is a time dependent mechanism and it is known to be exponentially dependent on temperature, supply voltage and *NBTI-duty-cycle*. The Reaction-Diffusion model [16] represents the generally accepted model of threshold voltage shift based on experimental observations. This work is focused on *NBTI-duty-cycle* reduction to mitigate the NBTI impact, since the duty-cycle usage of the target devices has direct influence on the NBTI degradation. This is well remarked from both theoretically and experimentally results in this research area, as well as explained by the long-term model reported as closed-form in Equation 1 [17]:

$$|\Delta V_{th}| \approx \left(\frac{\sqrt{K_v^2 \cdot T_{clk} \cdot \alpha}}{1 - \beta_t^{1/2n}} \right)^{2n} \tag{1}$$

where K_v is a parameter dependent on supply voltage and operating temperature, T_{clk} is clock period, α is the stress probability of PMOS devices, i.e. the *NBTI-duty-cycle*, β_t is dependent on temperature, while n is generally set to $1/6$ [18]. To this extent our methodology recovers NBTI degradation on buffers by reducing the *NBTI-duty-cycle* [7] using micro-architectural information and by exploiting power gating. Each buffer is enhanced with a header PMOS transistor that is responsible to cut the supply voltage when required. The sleep transistors act as switches that toggle supply voltage when required (from V_{DD} to *virtual* V_{DD}). More details on the design of header PMOS transistors, as well as on the influence of aging on sleep transistor can be found in [19].

Starting from the baseline NoC, let us consider a single upstream and downstream routers pair for simplicity, where the upstream router sends flits to the input buffers of the corresponding downstream one. Two observations are valid for both the sensor-less and the sensor-wise methodologies and can also be generalized to each upstream downstream routers pair in the NoC:

- *single flit per cycle* – a single flit can flow from each upstream downstream routers pair, thus at most an idle virtual channel is needed to store such flit if it belongs to a new packet. Note that we do not allow packet mixing in a single VC buffer;
- *traffic information exploitation* – the single idle VC can be switched-off based on traffic information, since the VC allocation in the downstream router happens in the upstream one. Upstream router holds all the information on new packets that want to go to the specific downstream router and have not a VC allocated yet on the downstream router. These new packets are eventually stored in the input ports of the upstream router. This way the upstream router needs an additional link to send the VC identifier to be left idle by the downstream router if new packets are waiting for the downstream router. It is worth noticing that we refer in this context to new packets as packets that are waiting in an upstream router input port, while they have no VC assigned, yet. Otherwise the downstream router can switch off all its idle VCs, since no new flit are expected. This way all the incoming flits to the downstream router input port will be stored in one of its already active VC.

B. Sensor-less Round-Robin Approach

This section details the sensor-less round-robin policy, i.e. *rr-no-sensor*, to minimize the NBTI stress period on the most degraded virtual channel buffer.

Starting from the considered baseline router and the additional information we can exploit considering the NoC architecture, the best approach that we can cast without NBTI sensors is to cyclically recover all the virtual channel buffers following a round-robin policy.

Algorithm 1 provides a detailed view of the round-robin sensor-less policy that is used as reference model throughout this paper. Assume an upstream downstream router pair, where the upstream router is in charge of the virtual channel allocation (VA) stage for the downstream input port, i.e. an hypothetical *outport x*. The upstream router considers the *output VC state* on the corresponding downstream router input port to know the state on the downstream VCs, i.e. the *out_vc_state* that is an input of the algorithm. Moreover, it is aware of the VCs that are storing packets which must be routed to *outport x*. In particular, the *is_new_traffic_outport_x()* function is an input to Algorithm 1 and returns 1 if at least a new packet wants to go to the *outport x* and has no VC allocated yet. The upstream router has an *active_candidate* VC identifier, that is changed cyclically on a time basis. The *active_candidate* VC identifier represents the first VC the upstream router tries to use in the case of a new packet allocation. The output of Algorithm 1 is a VC identifier that must be left idle by the downstream router, i.e. *active vc*, and an enable signal that asserts to the downstream router, whether or not the VC identifier is valid, i.e. *enable*. At each cycle at most one new virtual channel could be required in the downstream router, to store flits from a new packet, thus the *active_candidate* bit signals which is the first virtual channel that must be considered for use if a new virtual channel is needed. If no new packets require VA stage, the upstream router de-asserts the *enable* signal, thus the downstream router can recover all of its idle VCs on the corresponding input port, as detailed in lines 4-7.

Algorithm 1: The *rr-no-sensor* pre-VA stage for each upstream router.

```
   Input: out_vc_state, active_candidate_vc, is_new_traffic_outport_x()
   Output: enable, active_vc
1  enable ← 0;
2  active_candidate = get_vc_candidate();
3  boolTraffic = is_new_traffic_outport_x();
4  if not boolTraffic then
5      enable ← 0;
6      active_vc ← active_candidate;
7      return;
8  offset_vc ← active_candidate;
9  foreach iter ∈ (1..num_vcs) do
10     if is_idle(offset_vc) or is_recovery(offset_vc) then
11         set_idle(offset_vc);
12         enable ← 1;
13         active_vc ← offset_vc;
14         return;
15     offset_vc++;
16     if offset_vc ≥ num_vc then
17         offset_vc ← 0;
```

C. Sensor-wise Approach

This section details the logic modifications implemented in the baseline on-chip network to exploit the sensor-wise NBTI policy that is presented on a couple of routers only for the sake of clarity and conciseness.

The main idea of this proposal is the decoupling of the measurement and the selection steps. In particular, for each upstream downstream routers pair, the NBTI degradation is evaluated in the downstream router by means of sensors, since it physically hosts the VC buffers. On the other hand, the selection of the downstream router VCs that can be recovered is in charge to the upstream one, since it has all the required traffic information and performs the VA stage for the VCs in the downstream router.

The baseline on-chip network and the modified NBTI aware NoC are proposed in Figure 1A and 1B respectively, considering two routers – *Router A* and *Router B* – and a single communication direction (from *A* to *B*), thus *Router A* is the upstream router, while *Router B* is the downstream one. We consider a four input/output port router model (North, South, West, East) with four virtual channel buffers for each input unit. At each time a buffer can store one or more flits of a packet, or can be idle from the network point of view, while it can be considered always stressed from the NBTI point of view. We considers a buffer recovered if it is switched-off. The proposed

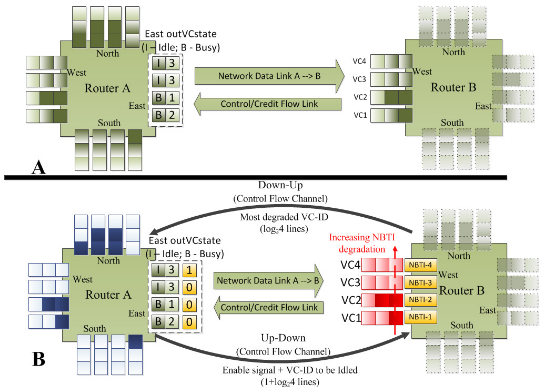

Fig. 1. Baseline (A) and sensor-wise NBTI-aware (B) micro-architectures.

strategy tries to reduce the *NBTI-duty-cycle* that is one of the main NBTI degradation factors. We focus on the west input port of *Router B* (red buffers in Figure 1B) and the corresponding East output port of *Router A*. It is worth to notice that the discussion is valid for any other input and output ports pair in the NoC. Figure 1B reports the *output VC state*, i.e. *outVCstate*, for the East output port in Router A instead of the input buffers to enhance the presentation of the methodology. In the baseline router the *output VC state* contains useful information that track the state of the corresponding virtual channel buffers in the west input port of *Router B*, i.e. the state of the corresponding VC and the available flit slots. The yellow blocks in the East *output VC state* represent the additional logic to support the NBTI sensor-less methodology. In particular, we require an additional bit for each VC associated to the *output VC state* as the *most_degraded* marker. One-hot encoding for *most_degraded* VC is possible, since there is only a single *most_degraded* VC at each time. Algorithm 2 details the recovery policy the upstream router uses to select a single virtual channel to be idle in the downstream router, if at least a new packet needs to traverse the downstream router. Otherwise it signals the downstream router to recover all its idle VC buffers.

Algorithm 2: The *sensor-wise* pre-VA stage for Router A East output port.

```
   Input: out_vc_state, most_degraded_vc, is_new_traffic_East_outport()
   Output: enable, idle_vc
1  enable ← 0;
2  idle_vc ← -1;
3  count_idle ← 0;
4  boolTraffic = is_new_traffic_East_outport();
5  foreach iter ∈ (1..num_vcs) do
6      if is_idle(offset_vc) or is_recovery(offset_vc) then
7          set_idle(offset_vc);
8          count_idle++;
9  if is_idle(most_degraded_vc) and count_idle> boolTraffic then
10     set_recovery(most_degraded_vc);
11     count_idle −−;
12 foreach iter_vc ∈ (1..num_vcs) do
13     idle_vc ← iter_vc;
14     if is_idle(iter_vc) and count_idle> boolTraffic then
15         set_recovery(iter_vc);
16         count_idle−−;
17 if boolTraffic then
18     enable ← 1;
```

The algorithm exploits three information sources. First, it knows the number of incoming packets that wants to go to the East output port of the router itself, i.e. a subset of the packets stored in the blue buffers of *Router A*. Second, the algorithm knows the most degraded VC in the west input port of *Router B*, i.e. one of the red VC buffers. This information comes directly from *Router B*. Last, it knows the actual state of west input port VCs in the *Router B*, i.e. through the output port VC state information. The first step of the algorithm restore to idle all the recovered VCs in lines 5-8, since we have to recover the *most_degraded_vc* first. Then, we recover the *most_degraded_vc* if it is idle and there is at least another idle VC in case of new packets for this output port – lines 9-11 –. The last part of the algorithm tries to recover all the idle VCs according to the boolean value returned by the *is_new_traffic_East_output_port* function, i.e. 1 if there is at least a packet that wants to go through the East output port in Router A and has not assigned a VC in the downstream router yet. It is worth to note, that for each iteration a single idle VC is selected (line 12), while the real decision to leave the VC idle is taken at lines 17-18 through the *enable* signal. An idle VC value is always present at the end of the algorithm, while the VC is actually left idle if and only if the *enable* signal is asserted.

For each downstream router, i.e. *Router B* in this example, we require an NBTI sensor [20] for each VC buffer to monitor the V_{th} degradation, so to select the most degraded VC buffer.

This is the only modification needed in the downstream router. Until now, we focused on the recovery selection policy that happens in the upstream router and the NBTI evaluation, that is in charge to the downstream router. The final step is to insert 2 additional control flow links to allow information exchange between each downstream-upstream routers pair.

The *Up_Down* link in Figure 1 allows to send a single VC identifier (VC-ID) from the upstream to the downstream router. The VC-ID is the VC in the downstream router that must be left idle to eventually store flits from a new incoming packet. In this way the link requires $log_2(num\_vc)$ lines, where num_vc is the actual number of VCs per input port. Moreover, an additional line is reserved for the *enable* signal that asserts the validity of VC-ID lines, since a valid VC-ID is always present on these lines.

The *Down_Up* link in Figure 1 allows to signal the upstream router the most degraded VC in the downstream router. The link requires $log_2(num\_vc)$ lines and there is no need for an additional enable line, since a most degraded VC is always present. This information steer the changes of the *most_degraded* bit in the corresponding *out VC state* of the upstream router.

D. Coarse Grain Sensor-wise Overhead

The changes due to the *sensor-wise* methodology impact the area of the baseline router. This section discusses the feasibility of our proposal from the area overhead point of view, that is under 4% of the baseline reference NoC. We consider data from a 45nm synthesizable NBTI sensor presented in [20], that is a sensor particularly suitable for high volume placing into a micro-architectural design. Moreover, we use ORION2.0 [21] to extract router and link area information for a 45nm technology node. We obtain an area overhead due to all NBTI sensors, i.e. 16 sensors = 4 input-ports x 4 VCs_per_port (one sensor per VC buffer), of 3.25% on the entire router considering 64 bits per flit, 4 VCs per input port and 4 flits per buffer. Moreover, we have an area overhead due to additional *up_down* and *down_up* links of 3.8% with respect to a single 64bits data link. Moreover, we integrate both the Algorithm 2 and the comparator logic in the upstream and downstream routers, respectively in the NetMaker [22], that is a library of synthesizable NoC components. The final design has been synthesized using Cadence Encounter compiler providing a negligible area overhead of the additional logic.

TABLE I
EXPERIMENTAL SETUP: PROCESSOR AND ROUTER MICRO-ARCHITECTURES AND TECHNOLOGY PARAMETERS.

Processor core	1GHz, out-of-order Alpha core
Int-ALU	4 integer ALU functional units
Int-Mult/Div	4 integer multiply/divide functional units
FP-Mult/Div	4 floating-point multiply/divide functional units
L1 cache	64kB 2-way set assoc. split I/D, 2 cycles latency
L2 cache	512KB per bank, 8-way associative
Coherence Prot.	MOESI token (for real traffic)
Router	3-stage wormhole switched with 32b link width
	2/4 virtual channels for each virtual network
	2/6 virtual networks (Garnet network [15])
	- data virtual network used for real traffic simulations
	- instruction virtual network used for synthetic traffic simulations
Topology	2D-mesh, based on Tilera iMesh network [23]
	for link width and NoC frequency (@1GHz)
Technology	$V_{th} = 0.160$ at 32nm and $V_{th} = 0.180$ 45nm, Vdd=1.2V

IV. RESULTS

This section details *NBTI-duty-cycle* reduction, considering process variation issues as well. Experimental setup is presented in Section IV-A, while results for synthetic traffic patterns and real scenarios are discussed in Section IV-B and Section IV-C.

A. Experimental setup

We performed cycle-accurate simulations using GEM5 performance simulator and Garnet network model [15] considering a 3-stages pipelined routers and 2D-mesh multi-core architectures. The multi-core architecture is composed of tiles. Each tile in the 2D-mesh is composed of an out-of-order processor based on Alpha-21264 ISA, private L1 cache and shared distributed L2 cache banks, and a memory controller. Table I summarizes the main architectural setup and technology parameters of the simulated multi-core architecture.

We also consider process variation as a true driver of variability in scaled technologies. Process variation is a combination of random effects and systematic effects [24], and can impact both die-to-die and within-die parameters. In this work we focus on within-die process variation, and assume for simplicity the impact of die-to-die variation to be constant in the same chip [13]. Process variability manifests itself as a divergence of sensible design parameters (e.g., initial threshold voltage V_{th}) from their nominal values. For these reasons, we assume the same initial V_{th} for each VC, and equal to the highest V_{th}, namely that of the most degraded PMOS transistor in the buffer. We then associate a PMOS transistor to each virtual channel buffer of each router; each modeled PMOS transistor has its own starting V_{th}, that has been extracted from a Gaussian distribution with absolute average value of 0.180 Volt for 45nm technology, and a standard deviation equal to 0.005 [25].

B. Synthetic Results

This section reports the results obtained considering uniform traffic patterns, on both 4-core and 16-core 2D-mesh architectures, with 2 and 4 virtual channels per input port and considering varying injection rates of 0.1, 0.2 and 0.3 flits/cycles/port. To collect representative statistics, we simulated each scenario for $30 * 10^6$ cycles, while at the end of the simulation we sampled the *NBTI-duty-cycle* values for each virtual channel. The network reaches the steady state after 6 and 9 million cycles for 4-core and 16-core, respectively. Each result is sampled from the upper left-most router on its east input port due to symmetry in the topology. Results for 4- and 16-cores scenarios are reported in Table III considering 2 virtual channels per input port, while the same 4- and 16- architectures results using 4 virtual channels per input port are reported in Table II.

Table III and II share the same format, where the first and the second columns report the identifier of the simulated scenario and the most degraded virtual channel buffer identifier for the simulated

TABLE II
NBTI-DUTY-CYCLE (%) FOR ALL THE VCS USING *rr-no-sensor*, *sensor-wise-no-traffic* AND *sensor-wise* POLICIES. RESULTS FOR 4-CORES AND 16-CORES WITH VARIABLE INJECTION RATE AND 4 VCS.

Scenario	MD	rr-no-sensor				sensor-wise-no-traffic				sensor-wise				Gap
(4 VCs)	VC	VC0	VC1	VC2	VC3	VC0	VC1	VC2	VC3	VC0	VC1	VC2	VC3	(rr-no-sensor − sensor-wise)
4core-inj0.10	1	11.9%	12.1%	12.5%	11.7%	3.8%	0.3%	100%	26.2%	1.5%	0.1%	9.9%	36.8%	$11.7 - 0.1 = 11.6\%$
4core-inj0.20	0	19.8%	19.9%	20.2%	20.0%	2.2%	12.8%	45.0%	100%	1.2%	6.6%	22.6%	49.3%	$20.2 - 1.2 = 19.0\%$
4core-inj0.30	3	27.7%	27.8%	27.8%	27.8%	24.3%	58.6%	100%	6.6%	14.5%	34.5%	58.1%	4.0%	$27.8 - 4.0 = 23.8\%$
16core-inj0.10	1	17.6%	17.3%	17.5%	17.4%	10.9%	2.0%	40.9%	100%	5.2%	0.9%	18.9%	44.8	$17.3 - 0.9 = 16.4\%$
16core-inj0.20	1	31.6%	31.5%	31.6%	31.4%	31.1%	10.9%	63.6%	100%	19.6%	7.9%	39.3%	60.5%	$31.5 - 7.9 = 23.6\%$
16core-inj0.30	2	46.2	46.3%	46.2%	46.3%	51.6%	77.9%	26.7%	100%	37.6%	56.2%	19.6%	71.4%	$46.2 - 19.6 = 26.6\%$

architectures, i.e. *Scenario* and *MD* columns. It is worth to notice that the most degraded VC changes through different simulations due to the random sampling process that mimics process variation impact, as explained in Section IV-A. In particular we sampled a single set of PMOS V_{th} values for each pair $\{$*simulated architecture, traffic injection*$\}$, thus different policies use the same V_{th} values on the same simulated architecture and traffic level for consistency purposes. Both Table III and II report three multi-columns, one for each evaluated policy. In particular, we use the round-robin sensor-less policy, i.e. *rr-no-sensor*, as reference and it is reported in the first multi-column of each table. This policy exploits traffic information available in the upstream router for recovery purposes, as discussed in Section III. The second multi-column reports data for the *sensor-wise* policy that does not consider additional traffic information for recovery selection, i.e. *sensor-wise-no-traffic*. The last multi-column contains results for the *sensor-wise* policy considering traffic information for VC recovery purposes, i.e. *sensor-wise*, and represents the proposed of the paper. For each policy, i.e. multi-column, we report the *NBTI-duty-cycle* obtained at the end of the simulation for each VC. The last column of each table (*Gap*) reports the *NBTI-duty-cycle* difference between the reference round-robin sensor-less and our *sensor-wise* policies. Positive *Gap* values means *sensor-wise* behaves better than *rr-no-sensor* policy, i.e. the second obtains lower *NBTI-duty-cycle*.

We can draw two relevant conclusions from the reported data. First, the additional traffic information exploitation, all the multi-columns in Table III and Table II but the *sensor-wise-no-traffic*, introduced in Section III-B, allows for a great *NBTI-duty-cycle* reduction. Moreover, such reduction is independent from the size of the simulated architecture, i.e. 4- or 16-cores, and the traffic level. Moreover it can reduce the *NBTI-duty-cycle* on all the virtual channel even if they are not the most degraded. For example in the second line on Table III, VC1 is the most degraded virtual channel. The reported *NBTI-duty-cycle* for VC1 using *sensor-wise-no-traffic* is 49.5%, while the same VC has a lower *NBTI-duty-cycle* equal to 26.5% using the *sensor-wise* policy that exploits traffic information. The same trend can be observed on 16-core architecture, e.g. the fifth line in Table III. The most degraded VC (VC0) reports an *NBTI-duty-cycle* equal to 76.3% and 51.7% for the *sensor-wise-no-traffic* and *sensor-wise* respectively. Last, the least degraded VCs are subject to an higher NBTI-stress level using the *sensor-wise-no-traffic* policy, since an idle VC is always present waiting for a flit, even if no new packet is present in the upstream router.

The second observation is related to the effectiveness of the *sensor-wise* approach. Even if it can always obtain lower *NBTI-duty-cycle* than the *rr-no-sensor* policy, it is worth noticing that the *NBTI-duty-cycle* gap between *sensor-wise* and *rr-no-sensor* reduces with the traffic load, while increases with the number of virtual channels. The increase in the traffic load prevents the *sensor-wise* policy to employ VCs different from the most degraded one to steer new packets, since we have an high probability that all the VCs are busy at the same time. On the contrary, the *rr-no-sensor* approach is independent of this aspect, since new packets are assigned to VCs in a round-robin fashion. This aspect is verified considering the first three rows in Table III, where the injection rate is 0.1 0.2 and 0.3 respectively. The *Gap* column reports 13.4%, 12.8% and 9.5% for the three injection rates respectively. On the other hand this trend is not present considering the first three lines in Table II, where the *Gap* between *sensor-wise* and *rr-no-sensor* policies increases with the injection rate. However it is worth to notice that this behavior is due to the increased number of virtual channels, i.e. from 2 to 4 VCs, that allows to achieve a better control over the *NBTI-duty-cycle* for the *sensor-wise* approach, since the NoC is never congested, i.e. VC *NBTI-duty-cycle* is always under 60% considering the third line in Table II with injection rate 0.3 flits/cycle/port.

TABLE III
NBTI-DUTY-CYCLE (%) FOR ALL THE VCS USING *rr-no-sensor*, *sensor-wise-no-traffic* AND *sensor-wise* POLICIES. RESULTS FOR 4-CORES AND 16-CORES WITH VARIABLE INJECTION RATE AND 2 VCS.

Scenario	MD	rr-no-sensor		sensor-wise-no-traffic		sensor-wise		Gap
(2 VCs)	VC	VC0	VC1	VC0	VC1	VC0	VC1	[rr-no-sensor−sensor-wise]
4core-inj0.10	1	23.8%	23.8%	100%	27.5%	37.3%	10.4%	$23.8 - 10.4 = 13.4\%$
4core-inj0.20	1	39.3%	39.3%	100%	49.5%	52.1%	26.5%	$39.3 - 26.5 = 12.8\%$
4core-inj0.30	0	56.2%	56.5%	69.6%	100%	46.7%	65.9%	$56.2 - 46.7 = 9.5\%$
16core-inj0.10	0	33.4%	33.6%	43.6%	100%	20.1%	46.9%	$33.4 - 20.1 = 13.3\%$
16core-inj0.20	0	61.8%	61.6%	76.3%	100%	51.7%	71.8%	$61.8 - 51.7 = 10.1\%$
16core-inj0.30	1	72.9%	73.0%	100%	85.3%	80.7%	65.1%	$73.0 - 65.1 = 7.9\%$

C. Average Results for Real Traffic Scenarios

This section reports in average results obtained for 4- and 16-cores 2D-mesh topologies considering 2 VCs and a simulation time of $30 * 10^6$ cycles. We do not include the results for 4 VCs due to space limitation. We simulate each architecture scenario, i.e. for each combination of core number and virtual channel number, 10 times. For each iteration we randomly picked up a set of benchmarks, i.e. one for each core of the simulated architecture, from SPLASH2 and WCET benchmark suites. Moreover we maintain the initial voltage threshold values constant for the 10 iterations of the same architecture to guarantee results consistency even if different benchmark mixes where executed, i.e. for each scenario the most degraded VC is constant through all iterations. Last we reported both the average (*avg*) and standard deviation (*std*) *NBTI-duty-cycles* for each VC of the considered scenario. It is worth to notice that we report each router and its east or west input port for all the 4-core simulated architectures, while the *NBTI-duty-cycle* for the east input port of the main diagonal routers in the 16-core architectures are also provided.

The first column in Table IV identifies a specific router in the multi-core architecture and its considered east (*E*) or west (*W*) input port. The second column reports the most degraded VC (*MD VC*) for the scenario, due to the random V_{th} sampling process to mimic process variation impact on PMOS. Moreover, Table IV has two multi-columns reporting the *NBTI-duty-cycle* for both *rr-no-sensor* and the *sensor-wise* policies. Each multi-column reports all the VCs, while for each VC both the average and the standard deviation *NBTI-duty-cycle* on the 10 iterations are reported. The last column of each

TABLE IV
NBTI-DUTY-CYCLE (%) FOR ALL VCS USING THE *rr-no-sensor* AND THE *sensor-wise* POLICIES, CONSIDERING 4- AND 16-CORES WITH VARIABLE BENCHMARK MIXES AND 2 VCS. AVERAGES OVER 10 ITERATIONS.

Scenario (2 VCs)	MD VC	rr-no-sensor				sensor-wise				Gap
		VC0		VC1		VC0		VC1		
		avg	std	avg	std	avg	std	avg	std	avg
4c-r0-E	1	6.5%	11.3%	6.5%	11.4%	7.9%	13.5%	5.2%	9.1%	1.3%
4c-r1-W	0	12.8%	13.4%	12.8%	13.5%	12.3%	12.3%	13.4%	14.9%	0.5%
4c-r2-E	1	16.8%	13.9%	16.7%	13.5%	29.4%	25.8%	3.7%	2.9%	13.0%
4c-r3-W	1	17.2%	11.7%	17.9%	12.4%	32.6%	23.2%	2.4%	2.1%	15.5%
16c-r0-E	0	22.2%	10.5%	22.1%	10.4%	3.3%	2.3%	40.9%	19.0%	18.9%
16c-r5-E	0	25.5%	20.2%	25.5%	20.2%	21.2%	16.8%	29.8%	23.6%	4.3%
16c-r10-E	1	9.1%	5.8%	12.1%	5.8%	20.6%	8.5%	0.7%	0.5%	11.4%
16c-r15-E	0	15.7%	8.3%	16.7%	7.9%	1.7%	1.4%	30.8%	14.8%	14%

table details the averaged *NBTI-duty-cycle* difference between the *rr-no-sensor* and the *sensor-wise* policies, i.e. *Gap* column.

There are two main observations from the results. First, the *sensor-wise* policy always outperforms the *rr-no-sensor* approach to reduce the *NBTI-duty-cycle* on the most degraded virtual channel, i.e. all the entries in column *Gap* are positive. Moreover, this trend seems quite independent from the multi-core size or the number of available multi-core.

Second, we observe a strong stability of our *sensor-wise* approach, since the standard deviation on the *NBTI-duty-cycle* for the most degraded virtual channel is always smaller than the corresponding standard deviation on the *NBTI-duty-cycle* for the *rr-no-sensor* approach. This means that we can aggressively reduce the *NBTI-duty-cycle* with minor divergences. For example the third line in Table IV reports a 4core scenario considering the east input port on router 2, where VC1 is the most degraded VC. The *sensor-wise* policy achieves a 3.7% *NBTI-duty-cycle* with a standard deviation equal to 2.9 on VC1, while the *rr-no-sensor* ensures only 16.7% and 13.5% for both average and standard deviation *NBTI-duty-cycle*, respectively.

V. CONCLUSIONS

We presented a cooperative *sensor-wise* methodology to minimize the impact of NBTI degradation in the most degraded virtual channel buffers of each input port of the NoC routers reducing the NBTI stress period, i.e. *NBTI-duty-cycle*. The validation have been carried out using a cycle accurate simulator, where we integrated an accurate NBTI library to mimic NBTI sensors [7] providing NBTI degradation cycle accurate statistics [6]. The *sensor-wise* methodology has been tested considering both synthetic and real traffic pattern scenarios, against the round-robin sensor-less NBTI mitigation approach, that is the best possible strategy to recover the most degraded VC without any information from NBTI sensors. We can reduce the *NBTI-duty-cycle* improving the activity NBTI factor over the best round-robin strategy up to 26.6% and 18.9% considering synthetic and real traffic patterns, respectively. Moreover, we used the model presented in [7] to extract the real NBTI V_{th} (details are omitted due to lack of space). It can be observed a net NBTI mitigation (less V_{th} degradation) of the *sensor-wise* methodology of up to 54.2% with respect to the baseline NoC that does not account for NBTI.

Furthermore, this work explores the cooperation between each upstream and downstream router pairs to aggressively recovery VC buffers exploiting traffic information. To this extent, the discussed results show how the cooperation can reduce the *NBTI-duty-cycle* on the most degraded VC buffer up to 23%. Last, the proposed *sensor-wise* methodology exhibits a fairly acceptable area overhead below 5% of the considered baseline model.

ACKNOWLEDGMENTS

This research work is partially supported by European Community Seventh Framework Programme (FP7/2007-2013), under agreements no. 248716 (2PARMA project www.2parma.eu).

REFERENCES

[1] A. Banerjee, R. Mullins, and S. Moore, "A Power and Energy Exploration of Network-on-Chip Architectures," in *NOCS '07*. IEEE Computer Society, 2007, pp. 163–172.

[2] L. Peters, "Nbti: A growing threat to device reliability," 2004.

[3] S. Nassif, K. Bernstein, D. Frank, A. Gattiker, W. Haensch, B. Ji, E. Nowak, D. Pearson, and N. Rohrer, "High performance cmos variability in the 65nm regime and beyond," in *Electron Devices Meeting, IEDM IEEE International*, 2007, pp. 569 –571.

[4] C. Nicopoulos, S. Srinivasan, A. Yanamandra, D. Park, V. Narayanan, C. Das, and M. Irwin, "On the effects of process variation in network-on-chip architectures," *Dependable and Secure Computing, IEEE Transactions on*, vol. 7, no. 3, pp. 240 –254, 2010.

[5] S. Kumar, K. Kim, and S. Sapatnekar, "Impact of nbti on sram read stability and design for reliability," in *ISQED*, 2006.

[6] D. Zoni, S. Corbetta, and W. Fornaciari, "Hands: Heterogeneous architectures and networks-on-chip design and simulation," in *IEEE ISLPED'12 International Symposium on Low Power Electronics and Design, Redondo Beach, California, USA*, 2012, pp. 261–266.

[7] T.-B. Chan, J. Sartori, P. Gupta, and R. Kumar, "On the efficacy of nbti mitigation techniques," in *Design, Automation Test in Europe Conference Exhibition (DATE), 2011*, march 2011, pp. 1 –6.

[8] D. Bild, G. Bok, and R. Dick, "Minimization of nbti performance degradation using internal node control," in *DATE*, 2009.

[9] L. Li, Y. Zhang, J. Yang, and J. Zhao, "Proactive nbti mitigation for busy functional units in out-of-order microprocessors," in *DATE*, 2010.

[10] T. Siddiqua and S. Gurumurthi, "A multi-level approach to reduce the impact of nbti on processor functional units," in *Proceedings of the 20th symposium on Great lakes symposium on VLSI*, ser. GLSVLSI '10. New York, NY, USA: ACM, 2010, pp. 67–72.

[11] B. Li, L.-S. Peh, and P. Patra, "Impact of process and temperature variations on network-on-chip design exploration," in *NoCS Second ACM/IEEE International Symposium on*, 2008, pp. 117 –126.

[12] U. Ogras, R. Marculescu, and D. Marculescu, "Variation-adaptive feedback control for networks-on-chip with multiple clock domains," in *DAC 45th ACM/IEEE*, 2008, pp. 614 –619.

[13] X. Fu, T. Li, and J. Fortes, "Architecting reliable multi-core network-on-chip for small scale processing technology," in *Dependable Systems and Networks (DSN), IEEE/IFIP Int. Conf. on*, 2010, pp. 111 –120.

[14] A. Kodi, A. Sarathy, A. Louri, and J. Wang, "Adaptive inter-router links for low-power, area-efficient and reliable network-on-chip (noc) architectures," in *ASP-DAC*, 2009.

[15] N. Agarwal, T. Krishna, L.-S. Peh, and N. Jha, "Garnet: A detailed on-chip network model inside a full-system simulator," in *IEEE Performance Analysis of Systems and Software, ISPASS*, 2009.

[16] M. A. Alam and S. Mahapatra, "A comprehensive model of PMOS NBTI degradation," *Microelectronics Reliability*, 2005.

[17] S. Bhardwaj, W. Wang, R. Vattikonda, Y. Cao, and S. Vrudhula, "Predictive modeling of the nbti effect for reliable design," in *Custom Integrated Circuits Conference. CICC '06.*

[18] A. Krishnan, C. Chancellor, S. Chakravarthi, P. Nicollian, V. Reddy, A. Varghese, R. Khamankar, and S. Krishnan, "Material dependence of hydrogen diffusion: implications for nbti degradation," in *Electron Devices Meeting, 2005. IEDM Technical Digest. IEEE International*.

[19] M.-C. Lee, Y.-G. Chen, D.-K. Huang, and S.-C. Chang, "Nbti-aware power gating design," in *Design Automation Conference (ASP-DAC), 2011 16th Asia and South Pacific*, jan. 2011, pp. 609 –614.

[20] P. Singh, E. Karl, D. Sylvester, and D. Blaauw, "Dynamic nbti management using a 45 nm multi-degradation sensor," *Circuits and Systems I, IEEE Transactions on*, vol. 58, no. 9, pp. 2026 –2037, 2011.

[21] A. Kahng, B. Li, L.-S. Peh, and K. Samadi, "Orion 2.0: A fast and accurate noc power and area model for early-stage design space exploration," in *DATE '09.*, 2009, pp. 423 –428.

[22] NetMaker, "http://www-dyn.cl.cam.ac.uk/ rdm34/wiki."

[23] D. Wentzlaff, P. Griffin, H. Hoffmann, L. Bao, B. Edwards, C. Ramey, M. Mattina, C.-C. Miao, J. Brown, and A. Agarwal, "On-chip interconnection architecture of the tile processor," *Micro, IEEE*, 2007.

[24] M. Alam, K. Kang, B. Paul, and K. Roy, "Reliability- and process-variation aware design of vlsi circuits," in *Physical and Failure Analysis of Integrated Circuits. IPFA 2007. 14th International Symposium on the*.

[25] K. Agarwal and S. Nassif, "Characterizing process variation in nanometer cmos," in *44th ACM/IEEE Design Automation Conference.*, 2007.