VAWOM: Temperature and Process Variation Aware WearOut Management in 3D Multicore Architecture

Hossein Tajik[1], Houman Homayoun[2], Nikil Dutt[1]
[1]Center for Embedded Computer Systems, University of California Irvine
[2]Department of Electrical and Computer Engineering, George Mason University
{tajikh, dutt}@uci.edu, hhomayou@gmu.edu

ABSTRACT
Three dimensional (3D) integration attempts to address challenges and limitations of new technologies such as interconnect delay and power consumption. However, high power density and increased temperature in 3D architectures accelerate wearout failure mechanisms such as Negative Bias Temperature Instability (NBTI). In this paper we present VAWOM (Variation Aware WearOut Management), an approach that reduces the NBTI effect by exploiting temperature and process variation in 3D architectures. We demonstrate the efficacy of VAWOM on a two-layer 3D architecture with 4x4 cores on the first layer and 4x4 last level caches on the second layer, and show that VAWOM reduces NBTI induced threshold voltage degradation by 30% with only a small degradation in performance.

Categories and Subject Descriptors
B.3.2 [**Design Style**]: Cache Memories; B.3.m [**Miscellaneous**]

General Terms
Algorithms, Management, Design, Reliability.

Keywords
Wearout, NBTI, 3D Integration, Variation.

1. INTRODUCTION
Advances in technology and reduction in feature sizes have made transistors faster and also allow larger die sizes that permit the designer to exploit many computational cores for parallel computing. Due to the increased complexity in newer technologies, interconnection power consumption and delay are becoming critical challenges that require new methodologies. 3D integration is a new manufacturing technology that reduces power consumption and increases the speed of interconnections, while improving packaging density [1]. In a 3D architecture, more than one layer of electronic circuits are integrated in a single chip. Vertically stacked dies are connected by TSVs (Through Silicon Vias) which leads to shorter and faster connections.

However, 3D integration does not come free. In fact higher performance and considerable reduction in energy consumption enabled by this technology, comes with the cost of higher power density and increased heat conduction paths [2][36]. As a result, the chip's temperature is much higher in 3D architecture compared to 2D architectures. Increased cooling cost, higher probability of timing errors, physical damages, and lifetime reduction are just a few of many consequences due to this higher power density.

Such operational conditions in 3D processors put more stress on the circuit and activate failure mechanisms such as Negative Bias Temperature Instability (NBTI) [3]. NBTI is one of the most important wearout mechanisms in PMOS transistors [4] [5] which gradually adds delay to transistors and leads to timing errors. Studies have shown that NBTI is highly dependent on the operating temperature [6].

The average temperature in a 3D architecture is much higher than for 2D architectures and temperature of vertically adjacent components in different layers are highly correlated [7][28][37]. Long conduction paths for layers farther from the heat-sink make them hotter (from this point we assume that upper layers are farther from the heat-sink). Placing cache banks in upper layers helps to reduce thermal hotspots, due to low power density in cache banks. The higher temperature of a 3D architecture, specifically in the layer most distant from the heatsink, has a significant effect in accelerating NBTI degradation.

The common architecture of a 3D processor, which stacks cache vertically on top of the logic, makes cache temperature dependent on the workload running in the logic underneath. Therefore, contrary to a 2D architecture, spatial temperature differences in cache banks could be very high, depending on the underlying logic power density.

Moreover, new technologies create process and operational variations that impose major challenges to processor design [8] [9]. Threshold voltage and other types of variation lead to large performance variation between different device components. Frequency and cache access time variation exist in new technologies which make some cores or cache banks more prone to wearout failure [13].

In this paper, we present VAWOM (Variation Aware WearOut Management), an approach that reduces wear-out induced by NBTI in cache banks and cores of a 3D architecture, using temperature and process variation aware schemes. VAWOM sacrifices some banks in the cache layer to proactively perform wearout recovery. During recovery, the SRAM cache and the associated router is deactivated. By exploiting task migration in the core layer, VAWOM balances the temperature in cores and cache memories to mitigate NBTI effects in the cores and the cache banks.

To the best of our knowledge VAWOM is the first work that highlights the large threshold voltage variation over time due to high power density in 3D architecture and investigates activity migration in the core layer and wearout recovery mechanism in cache layer to mitigate this variation. VAWOM makes three major contributions:

- Demonstrates the large thermal and threshold voltage variation in 3D cache, unlike 2D cache architectures.
- Presents novel activity migration and proactive recovery mechanisms that exploit thermal variation in 3D architecture to mitigate wearout effects in multiple stacked layers in 3D architecture. VAWOM balances wearout in different parts of a 3D architecture.

DAC'13 May 29 – June 7 2013, Austin, TX, USA.

- Reduces wearout in 3D architectures by exploiting frequency variation in cores and access time variability in the cache banks. VAWOM balances wearout across multiple layers by imposing less stress on the slower cores and cache banks.

2. Related Work

For prolonging the lifetime of digital circuits in the presence of aging mechanisms such as NBTI, several techniques have been proposed at different layers of abstraction. At the circuit level, the most effective method is using delay guard-bands [10], where a delay margin is added to the nominal design. Adding a large guard band negatively affects the performance of the circuit. At the system level, DVS has been used to speed up the circuit or reduce the stress condition [39]. Some works propose to use redundancy in functional units, caches, or cores for mitigation of aging in both reactive and proactive manner [14] [15] [16]. Reducing duty cycle (e.g., with bit flipping) has also been proposed to decrease the effects of NBTI [4] [17]. [11] uses frequency variation to hide aging in processors by a careful scheduling of hot jobs to fast cores. This technique negatively affects cache banks in the 3D architecture (as we'll discuss later in Section 4).

At the architecture level, in [18] the authors proposed to balance workloads among active cores. NBTI aware Dynamic Instruction Scheduling was proposed by [19] which distributes tasks over functional units to balance aging. Using BIST, [20] tries to monitor NBTI degradation in SRAM cells and reconfigures cache arrays.

These methods do not consider cache banks and large process variation in circuits. Unlike VAWOM, none of these works consider temperature variation between cache banks in 3D architecture. Moreover, the cache access time variation and interplay of variation-aware techniques in different layers are not considered in these previous efforts.

3. Preliminaries (NBTI)

Negative Bias Temperature Instability (NBTI) is one of the important wearout mechanisms that mostly affects PMOS transistors. Based on transistor bias voltage, NBTI has two modes: stress and recovery. When gate-source voltage is negative, the transistor is under stress; this increases the threshold voltage and leads to performance degradation (device aging). A cumulative performance degradation results in timing errors after a sufficiently long stress period. In the recovery mode (PMOS biased with positive voltage), some threshold voltage increase can be compensated [12]. We use the well-known Reaction-Diffusion (R-D) model for measuring threshold voltage shift when applying VAWOM. In this model, Si-H bonds are dissociated under negative gate-source bias by positive holes from channel (stress mode) and make interface traps and H atoms. Traps formed at the interface of gate oxide, cause increase in threshold voltage of PMOS transistor [12]:

$$\Delta V_{th} = q\, N_{IT}/C_{ox} \quad (1)$$

N_{IT} is number of interface traps, q is basic charge and C_{ox} is oxide capacitance.

When gate-source voltage becomes positive, some of the hydrogen atoms diffuse back to the gate oxide interface and anneal some of the broken bonds. This phenomenon reduces the number of interface traps and compensates threshold voltage shift partially (recovery mode).

3.1 Stress

Negative voltage on the gate-source of PMOS transistor creates some interface traps. Equation (2) is a differential expression that describes formation of H, H2 and interface traps in the R-D model [5]:

$$\frac{N_{IT}}{t} - \frac{\delta k_H \left(k_f N_0 - \frac{N_{IT}}{t}\right)^2}{{k_r}^2 {N_{IT}}^2} + \frac{\delta k_{H2} N_{IT}}{\sqrt{6 D_{H2} t}} = 0\,, \quad (2)$$

where t is time, k_f is Si–H bond-breaking rate, k_r is Si–H bond-annealing rate, and N_0 is the initial bond density available before stress. k_H, k_{H2}, D_{H2}, and δ are generation rate of H_2, dissociation rate of H_2, diffusion rate of H_2, and interfacial thickness (~1−2 Å) respectively.

In different time slots, Equation (2) can be reduced to different analytical solutions [5]. A considerable amount of degradation occurs in the primary stages of stress mode. In the long stress time, threshold voltage degradation rate decreases due to the small time exponent factor. Initial bond density (N_0) is affected by operating voltage and initial threshold voltage. Reducing gate-source voltage or increasing threshold voltage slows down the degradation [22].

3.2 Recovery

A positive voltage on the gate-source of PMOS transistor recovers some interface traps. From [23], if at the beginning of the recovery phase (time t_0), N_{IT} is equal to $N_{IT}^{(0)}$ and $\xi = 1/2$, the number of interface traps at the end of the recovery phase (time t) equals:

$$N_{IT} = N_{IT}^{(0)}[1 - (\xi t/t_0)^{1/2}/(1 + t/t_0)^{1/2}] \quad (3)$$

A large portion of recovery occurs in the first few seconds after beginning of the recovery.

3.3 Temperature Dependence of NBTI

The operating temperature changes NBTI sub-process rates such as diffusion rate, conversion rate, and reaction rates during stress time. In general, higher temperature exacerbates degradation due to NBTI. NBTI thermal dependence and direct effect of temperature on sub-process rates have been investigated and formulated in several recent works [6] [24].

4. VAWOM

4.1 A. Baseline Configuration

To illustrate VAWOM, we assume an exemplar baseline 3D floorplan[1], where a single cache layer stacks on top of a core layer (Figure 1). Our studied 3D architecture has 16 dual issue cores all in one layer. To avoid thermal build up in the core layer as a result of 3D stacking, we place the core layer close to the heatsink while the cache layer is placed farther from the heatsink. In the bottom layer (Layer 1), 16 core routers are connected through a mesh topology network. In the upper layer (Layer 2), 16 shared last level cache banks are connected via 16 routers connected in a mesh topology. The baseline configuration in Figure 1 does not show inter-layer interconnections (between routers).

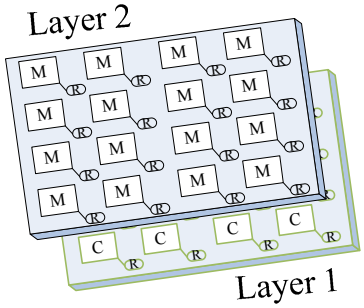

Figure 1. Baseline Configuration (C= Core, M = cache Memory)

[1] Of course our method could be applied to any 3D architecture.

4.2 VAWOM Overview

VAWOM deploys a proactive recovery method to manage temperature and process variation induced wearout. VAWOM uses several temperature and variation aware policies for activity migration and proactive recovery. These policies take into account device degradation (aging) in different layers, including cache bank access time degradation and core frequency degradation.

VAWOM runs periodically and in each run, two types of system change are performed. First, *activity migration in the core layer* with respect to NBTI thermal dependence improves the lifetime reliability of the cache banks and the cores. Second, *proactive recovery in the cache layer* improves cache lifetime, since aging is a more intense problem in the layers farther from the heatsink. In each run, VAWOM decides which cache bank should be placed in the recovery mode (called "recovery bank"), moves data from the recovery bank to the previously disabled/recovered cache bank, updates routing information, and disables new recovery bank. We first describe the experimental framework and the assumptions before describing VAWOM policies.

Unlike 2D architecture, temperature variation between different cache banks is noticeable in the 3D architecture. To better understand this characteristic, we study thermal variation in 2D and 3D architectures running with the same benchmark. We used Hotspot 5.0 for thermal estimation [25]. We observe large temperature variations of 25℃ across various cache banks in 3D. This is noticeably higher that the 4℃ thermal variation in the 2D design. Furthermore, cache temperature in the 3D architecture is about 24℃ higher than the 2D architecture. Thus, SRAM cache in the 3D architecture suffers from NBTI degradation significantly more than the 2D architectures.

In the baseline configuration without any wearout management technique, all of the cache banks are active and they are constantly under stress. This steady stress increases threshold voltage constantly over time. For compensating this degradation, an effective solution is to put cache blocks into the recovery mode. [15] proposed a circuit technique to put SRAM arrays of a cache bank in the recovery mode. VAWOM deploys a proactive recovery method to fight back the wearout in cache banks and puts whole cache bank in the recovery mode.

Furthermore, studies show that temperature has a significant effect on NBTI. Data in [24] indicates that in just 1000 seconds, threshold voltage increases by 10 mV as operating temperature increases from 75 ℃ to 125 ℃. As the total power consumed in 3D architecture is significantly dominated by the core power, dynamic thermal management techniques to reduce overall temperature is more effective if applied to the core layer rather than the cache layer. In all of these thermal management techniques, contrary to VAWOM, wearout is not considered and the only goal is to reduce the temperature.

NBTI is dependent on many different parameters such as supply voltage, switching activity, temperature, process variation etc. We assume a fixed supply voltage for each part of the architecture.

Variation in threshold voltage and channel length of transistors leads to gate delay variation and therefore critical path delay variation between different cores. Therefore, different cores have different frequencies. To estimate this frequency variation we used the method described in [8].

It's determined empirically that gate delay is nearly linear with respect to the threshold voltage.

There are many SRAM cells in each cache bank. Variation in transistor delay also exists in cache banks and therefore different SRAM cells have different access times. The probability distribution of critical path delay for different SRAM cells is computed. The maximum value of SRAM cell access times in one bank is considered as the initial access time for the entire cache bank. NBTI-induced v_{th} shift increases the access time for each cache bank and we use the model introduced in [8] to compute the access time change in the cache banks.

VAWOM assumes a constant temperature for each component between two consecutive runs (because of the long period), even though right after the activity migration, cores and cache banks have a transient temperature.

From the perspective of NBTI recovery, most of the recovery happens in a short period of time: a few seconds after the start of the recovery [6]. Therefore, keeping a cache bank in the recovery mode for more than few seconds is not very beneficial.

VAWOM assumes all transistors in a single core or cache bank will have the same degradation and each core is equipped with a thermal sensor to obtain its temperature.

4.3 VAWOM's Temperature-Aware Scheme

Several algorithms for activity migration in a multicore architecture have been proposed [26]. VAWOM's temperature-aware scheme keeps the design and performance overhead of the activity migration low, by applying it very infrequently. VAWOM's temperature aware scheme examines two different algorithms for activity migration:

1) *Temperature-Balancing algorithm (T-balancing)*: in this algorithm, tasks are migrated due to their current temperature. VAWOM sorts cores based on their temperature and swap tasks as follows: swap hottest and coldest cores' tasks, swap second hottest and the second coldest cores' tasks and so on. This algorithm intuitively will help to balance the temperature (and the degradation) between different cores and caches.

2) *Aging-Balancing algorithm (A-Balancing)*: temperature-balancing algorithm and other activity migration algorithms do not consider chip degradation (aging) in the cores and the cache banks. Due to the idiosyncrasy of NBTI stress and recovery, it is important to consider degradation in the activity migration policy. In the aging-balancing algorithm, VAWOM assigns estimated coldest jobs to the most degraded core and estimated hottest job to the least degraded core. Frequency degradation is VAWOM's metric for aging in the core layer.

VAWOM's temperature-aware scheme uses one of the following policies for proactive recovery in the cache layer: 1) *Round-Robin (RR)*: where banks are put into recovery mode cyclically, or 2) *Most Degraded (Max)*, where the bank with the highest V_{th} degradation is put into the recovery mode. Appendix 9.3 presents more details about VAWOM's temperature-aware scheme.

4.4 VAWOM's Process Variation-Aware Scheme

The move to deep submicron process technologies have caused process variations to become a major challenge that must be dealt with during design of chip multiprocessors. In particular, device parameters such as threshold voltage, resistance per unit length, gate length and width suffer large variation. According to ITRS, threshold voltage variation is projected to become more and more severe with scaling due to stochastic dopant variation [38]. [29] reported that in 35nm technology, the standard deviation of threshold voltage is 30.28 mV. Process variation could be manifested as frequency variation between different processors or access time latency variation between different cache banks. In this work we exploit these variations to reduce wearout in memories and processors.

Process variation in processors can be used to extend system lifetime [30] [11]. [11] uses the frequency variation between

different cores and attempts to schedule tasks such that hotter tasks run on cores with higher frequency. This method results in temperature variation between cores. The cores with lower operating frequency will be colder and have smaller degradation, which is desirable in a 2D architecture. However, in a 3D architecture, higher temperature in core layer makes cache layer hotter and thus imposes more degradation in the upper levels. In contrast with 2D architecture, the temperature variation aggravates the NBTI effects in the upper layer. Therefore, using the scheme proposed in [11] aggravates lifetime in the upper layer and is not effective for 3D architectures.

VAWOM's variation-aware scheme considers two types of variation: frequency variation between different cores [11] and cache access time variation between different cache banks [31]. Similar to our temperature-aware scheme, VAWOM's process variation-aware scheme uses activity migration in the core layer and proactive recovery in the cache layer for improving chip lifetime. This is done by imposing less threshold voltage degradation on slower cores (with lower operating clock frequency) and slower caches (higher access latency). For activity migration, VAWOM's process variation-aware scheme follows two policies: migration of hotter tasks to 1) faster cores or 2) cores whose adjacent cache bank in the upper layer is faster (Figure 7 in Appendix 9.4 details these two policies). The first policy (core based policy) has a negative effect on the performance degradation (wearout) of the cache layer. In case a slow cache bank is vertically adjacent to a faster core, this policy will impact the slow cache bank temperature which could potentially result in access delay degradation and therefore makes it even slower. The second policy (cache based policy) may have a negative effect on the core layer because slow cores may be vertically adjacent to fast cache banks. Therefore it is important to study the tradeoff between these two policies and develop a policy that considers all the layers simultaneously. We call this policy "Interleaved Policy".

It's likely that a 3D architecture will have different components in different layers. For example (in our case), we have cores in one layer and cache banks in the other layer. NBTI reduces the speed of a circuit until occurrence of a timing failure. We should specify a threshold for each component to be operational. For example, if the access time of a cache bank exceeds acc_{th} we consider it as a failure and that cache bank can't be used anymore. Similarly, for a core, if the critical path delay exceeds cp_{th} we have a failure in the core and it's not operational anymore. To exploit the process and temperature variation in different components, we need a unified criterion to compare different components. Therefore we use the term Distance to Failure (DF) which computes the distance of current state to the failure state:

$$DF_{core}(i) = \frac{cp_{th} - \mathrm{cp}(i)}{cp_{th}} \quad (4)$$

$$DF_{cache}(i) = \frac{acc_{th} - \mathrm{acc}(i)}{acc_{th}} \quad (5)$$

cp(i) is the current critical path delay of core i, cp_{th} is the threshold critical path delay, acc(i) is the current access time of cache bank i, and acc_{th} is the threshold access time. By using DF (which is between 0 and 1), we have a unified metric for different components. As temperatures in vertically adjacent components are highly correlated, we just consider core layer temperature. *Temp(i)* is the temperature of *Task(i)* (the task running on the core *i)*. Figure 2 outlines the VAWOM which is run periodically and in each run tries to solve this problem:

Inputs:

Access time for caches, frequency for cores, cores temperature

Outputs:

1) New task mapping (task migration): for all tasks and cores $Core(i) \leftarrow Task(j)$

2) New disabled cache bank.

Objectives:

1) Maximize Minimum Distance to Failure of components after running VAWOM for certain time: This objective tries to postpone the first failure of the components as much as possible.

2) Reduce the number of Hotspots.

It's noteworthy that in this work we want to keep components operational as much as possible. Any other objective can be selected based on the context.

The optimal output would be obtained by brute-force search (examining every combination of task assignment and disabling cache bank). But this method needs examining $n! \times n$ different combinations which is infeasible for even small values of n. Consequently, we introduce a heuristic for running VAWOM.

```
1:  Inputs: Critical Path Delay: cp[1..n];  Cache access time: acc[1..n];
CoreTemp: T[1..n];  TaskHash: task_hash[T]→{1,...,n}
2:  For i = 1 to n:     ---<Compute Distance to failure>
3:        DF_core [i] = (cp_th − cp[i]) / cp_th
4:        DF_cache [i] = (acc_th − acc(i)) / acc_th
5:  A = Sort_ascending(DF_cache) ---<find the cache for disabling>
6:  Disable (A[0])
7:  Remove (DF_cache, A[0])
8:  For i = 1 to n:     ---<Group Vertically Adjacent Components >
9:      If method == min:
10:           DF[i] =min {DF_core [i], DF_cache[i]}
11:     If method == mean:
12:           DF[i] =mean {DF_core [i], DF_cache[i]}
13:  Sort_ascending(DF)
14:  Sort_ascending(T)
15:  For i = 1 to n:     ---<task assignment><avoiding hotspots>
16:      For j =1  to window_size  :
17:          Neighbors[j] = Number_of_assigned_neighbors(DF[j])
18:      max_i = index(max(Neighbors))
19:      Task[task_hash(T[i])] → Core[DF_hash(DF[max_i])]
20:      Remove(DF, DF[max_i])
21:  Run (benchmark)
22:  T ← new Temperature
23:  cp ←new_cp(core_model, T)    ---<update degradation>
24:  acc ←new_acc(cache_model, T)
```

Figure 2. Interleaved+Max policy, VAWOM's process variation-aware scheme.

We call the policy introduced in Figure 2 "Interleaved + Max" (which disables max degraded cache bank). In case of using Round Robin scheme for recovery in cache banks, we call the policy "Interleaved + RR". In case of not using the proactive recovery in the cache layer (not disabling any cache bank), we just use the proposed activity migration and call it "Interleaved".

Figure 2 outlines the interleaved heuristic as described below (numbers in parenthesis are referring to the related lines in the figure). First, we compute the Distance to Failure (DF) for different parts (*2-4*). Minimum DF in the cache banks is the most degraded cache and should be disabled in the next period (*5-6*). We remove this cache from the cache DF list (*7*). In the next step, the criteria for task migration should be applied (*8-20*). For doing task migration, the order of the task assignment is from the hottest task to the coldest task. We group all vertically adjacent components as a unit entity, and examine two different methods (*8-12*): first, we consider the minimum {DF_{core}[i], DF_{cache}[i]} as the representative of DF[i]; and second, we consider the average {DF_{core}[i], DF_{cache}[i]} as the representative of DF[i]. Our results showed that the first one outperforms the second one. In the result section we used the first method (minimum).

We sort the DF list and the Temperature list (*13-14*). If we simply assign the hottest task to the entity with the maximum DF, we may have some hot parts in one layer adjacent to each other which can increase the risk of creating hotspot. For dealing with this problem, instead of picking the maximum DF, we look at a window of entities with the maximum DFs (for example, 3 least degraded entities). Then for each of them, we count the number of neighbors in the architecture which have already been assigned a task. Since we start assigning the tasks from the hottest to the coldest, a larger number of already assigned neighbors means that the entity might be hotter in the future (*16-18*). Therefore, we assign the hottest task to the entity within the window, with the minimum number of already-assigned-task neighbors (*19*). Then we remove the hottest task and the selected entity from the lists (*20*) and do this operation for the remaining tasks. After that, with new configuration, we run the benchmarks (*21*), get the new temperatures (*22*), and compute the new frequency and access time for the cores and cache banks (*23-24*).

4.5 Performance Overhead

In all policies and mechanisms for wearout management, VAWOM maintains negligible performance overhead. This is mainly due to the fact that VAWOM is applied very infrequently. In all simulations, the largest performance loss is found to be approximately 3% (detailed results are in our Technical Report [40]). This small performance impact comes from the three following sources:

1- Disabling only a small part of the cache; one out of 16 banks, for proactive recovery.

2- Infrequent transferring of data from a cache bank to previously disabled cache bank.

3- Infrequent task migration in the core layer.

Observing these performance losses, we can state that our policies have negligible performance penalty and we have a lightweight wearout management scheme.

5. Experiments

5.1 Experimental Setup

The platform introduced in Figure 1 is implemented for evaluating VAWOM. In this platform, a cache layer with 16 shared banks is stacked on top of 16 cores in a 3D architecture. Cores and cache banks use routers to form a mesh topology for interconnection.

SMTSIM [32] is used for processor simulation. 64 multi-program workloads consisting of 16 threads each are used for simulation. The applications are selected from the SPEC2006 benchmark suites and results are averaged among these 64 workloads. For network simulation HORNET1.0 [33] is integrated into SMTSIM simulation. For core power estimation, we used McPAT [34]. Using the HORNET network simulator, power consumption of routers and number of accesses to cache banks are obtained. Cache bank power consumption is computed using Cacti [35]. Hotspot 5.01 [25] is used to compute the temperature of each cache bank and core. After obtaining the temperature of the cores and cache banks, threshold voltage degradation is computed using Equations (1)-(3) and the equations and parameters described in [5] [24].

We assumed an initial threshold voltage for different transistors with normal distribution ($\mu = 340mV$ and $\sigma = 3.2mV$) and a 6σ variance.

To determine the efficacy of VAWOM, the baseline configuration and different schemes of VAWOM are implemented and compared in the following section. First, we show the efficacy of temperature-aware schemes in reducing maximum degradation in the cache and core layer. Second we show the similarity of temperature-aware schemes in reducing average degradation. Finally, effectiveness of variation-aware scheme in the cache and core layer is investigated.

5.2 Experimental Results

Figure 3 shows the maximum access time increase among cache banks w.r.t nominal access time for different temperature-aware schemes introduced in Section 4.3. Moreover, Figure 3 shows the results for the maximum critical path delay (frequency) increase (decrease) in the core layer w.r.t nominal frequency. All results are normalized to the baseline configuration with no recovery support. Degradation is computed after 100000s of system work where the time step between each run of VAWOM is 10 seconds.

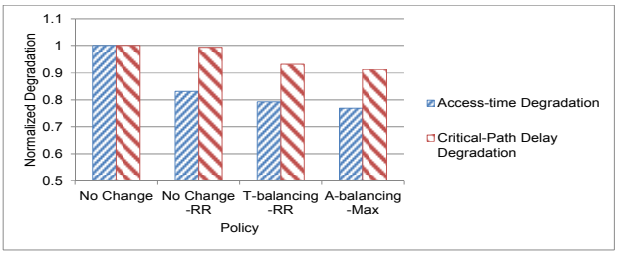

Figure 3. Maximum normalized access-time increase in caches and critical-path delay increase in cores after 100000s for temperature-aware schemes.

Because we use proactive recovery in the cache layer, degradation in this layer is much smaller than the degradation in the core layer. Using task migration in the core layer and proactive recovery in the cache layer (A-balancing - Max), we can achieve 23% improvement in the access time degradation in the cache layer. Also 8% improvement in the frequency degradation is obtained in the core layer. The A-balancing - Max scheme achieves the lowest degradation compared to the other schemes, as it applies the migration to the hottest core and applies the recovery to the most degraded cache bank. Our results show that change in the cache disabling policy (round-robin and max-degraded) affects the results at most 2%.

Figure 4 shows the average access time degradation in the cache layer w.r.t to nominal access time. VAWOM policies balance aging between different components. As a result, all VAWOM policies are almost uniformly effective in reducing average degradation effect.

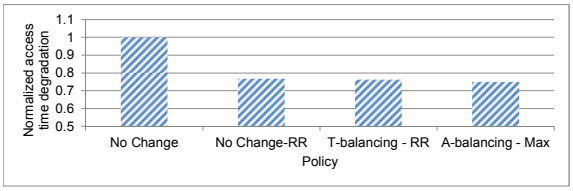

Figure 4. Average access time degradation in different cache banks after 100000s for temperature-aware scheme.

Figure 5 shows degradation improvement in variation-aware scheme introduced in Section 4.4 w.r.t nominal frequency and nominal access time. Note that in the four left bars no proactive recovery has been deployed. As shown in these figures, the interleaved schemes are more effective in lowering the degradation.

The core-based policy (Column 2) increases degradation in cache banks. Meanwhile, the cache-based policy (Column 3) aggravates degradation in cores (normalized degradation is bigger

than 1 in both cases). Using the interleaved policy (fourth Column), we are able to achieve the benefit of both policies and avoid their drawbacks (compare Columns 2, 3, and 4).

In the two right bars, effects of using proactive recovery methods are shown. In both studied methods (round-robin and max-degradation), more than 10% improvement is achieved. Our results indicate that degradation improvement is not very sensitive to cache disabling policy, as long as there is uniform distribution of the disabled cache.

Our results confirm that by considering both core and cache layers simultaneously for activity migration and using proactive recovery in the most degraded cache bank, we see improvement in the access time degradation by 31 % in the cache memories and improvement in frequency degradation by 16% in the core layer w.r.t to the nominal frequency. Furthermore, using the interleaved method, the maximum temperature could be reduced up to 4 ℃.

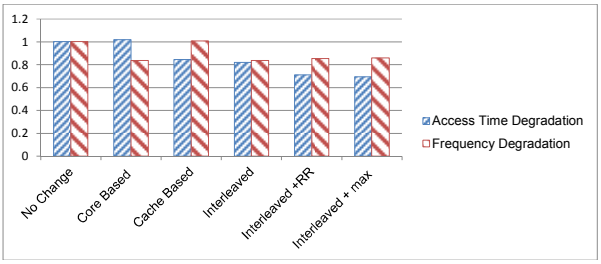

Figure 5. Normalized access-time degradation in the cache banks and frequency degradation in the core layer for variation-aware schemes.

6. CONCLUSION

In this paper, we proposed VAWOM, which takes into account the temperature and the process variation in 3D architecture and exploits activity migration mechanisms with respect to NBTI thermal dependence to improve the lifetime reliability of the system. We proposed and evaluated several temperature and process variation aware policies. We observed that by using very low overhead process variation-aware policies, threshold voltage degradation due to NBTI effect, could be reduced by 31% while maintaining performance at almost the same level. We believe VAWOM is the first effort to investigate simultaneous temperature and process variation aware wearout management for 3D multicore architecture. Future work will consider more than one recovery block and more sophisticated policies to mitigate the effects of wearout.

7. ACKNOWLEDGMENTS

This research was partially supported by NSF Variability Expedition Grant Number CCF-1029783.

8. REFERENCES

[1] J. Burns et al. Design, CAD and technology challenges for future processors: 3D perspectives. In *DAC*, 2011.
[2] A. K. Coskun et al. Dynamic thermal management in 3D multicore architectures. In *DATE*, 2009.
[3] J. Srinivasan et al. The Impact of Technology Scaling on Lifetime Reliability. In *DSN*, 2004.
[4] J. Abella et al. Penelope: The NBTI-Aware Processor. In *MICRO*, 2007.
[5] A. E. Islam et al. Recent Issues in Negative-Bias Temperature Instability: Initial Degradation, Field Dependence of Interface Trap Generation, Hole Trapping Effects, and Relaxation. *IEEE Trans. Electron Devices*, 2007.
[6] W. Wang et al. The Impact of NBTI Effect on Combinational Circuit: Modeling, Simulation, and Analysis. *IEEE Trans. Very Large Scale Integr. (VLSI) Syst,* 2010.
[7] X. Zhou et al. Thermal Management for 3D Processors via Task Scheduling. In *ICPP*, 2008.
[8] S. R. Sarangi et al. VARIUS: A Model of Process Variation and Resulting Timing Errors for Microarchitects. *IEEE Trans. Semicond. Manuf,* 2008.
[9] S. R. Nassif. Modeling and analysis of manufacturing variations. In *CICC*, 2001.
[10]S. V. Kumar et al. NBTI-Aware Synthesis of Digital Circuits. In *DAC*, 2007.
[11]A. Tiwari et al. Facelift: Hiding and slowing down aging in multicores. In *MICRO*, 2008.
[12]R. Vattikonda et al. Modeling and Minimization of PMOS NBTI Effect for Robust Nanometer Design. In *DAC*, 2006.
[13]A. Makhzan et al. Process Variation Aware Cache for Aggressive Voltage-Frequency Scaling. In *DATE*, 2009.
[14] J. Srinivasan et al. Exploiting structural duplication for lifetime reliability enhancement. In *ISCA*, 2005.
[15] J. Shin et al. A Proactive Wearout Recovery Approach for Exploiting Microarchitectural Redundancy to Extend Cache SRAM Lifetime. In *ISCA*, 2008.
[16] L. Huang et al. Characterizing the lifetime reliability of manycore processors with core-level redundancy. In *ICCAD*, 2010.
[17] E. Gunadi et al. Combating Aging with the Colt Duty Cycle Equalizer. In *MICRO*, 2010.
[18] J. Sun et al. NBTI Aware Workload Balancing in Multi-core Systems. In *ISQED*, 2009.
[19] T. Siddiqua et al. A Multi-Level Approach to Reduce the Impact of NBTI on Processor Functional Units. In *GLSVLSI*, 2010.
[20] F. Ahmed et al. Reliable cache design with on-chip monitoring of NBTI degradation in SRAM cells using BIST. In *VLSI Test Symposium*, 2010.
[21] A. Ricketts et al. Investigating the impact of NBTI on different power saving cache strategies. In *DATE*, 2010.
[22] T.-B. Chan et al. On the efficacy of NBTI mitigation techniques. In *DATE*, 2011.
[23] M. A. Alam et al. A comprehensive model of PMOS NBTI degradation. *Microelectronics Reliability,* 2005.
[24] Seyab et al. NBTI modeling in the framework of temperature variation. In *DATE*, 2010.
[25] W. Huang et al. HotSpot: Thermal Modeling for CMOS VLSI Systems. *IEEE Trans. Compon. Packag. Technol.,* 2005.
[26] C. Zhu et al. Three-Dimensional Chip-Multiprocessor Run-Time Thermal Management. *IEEE Trans. Computer Aided Design,* 2008.
[27] J. Wu. A Fault-Tolerant and Deadlock-Free Routing Protocol in 2D Meshes Based on Odd-Even Turn Model. *IEEE Transactions on Computers,* 2003.
[28] L. Tran et al. Heterogeneous Memory Management for 3D-DRAM and External DRAM with QoS. In *ASP-DAC*, 2013.
[29] D. Reid et al. Analysis of Threshold Voltage Distribution Due to Random Dopants: A 100□000-Sample 3-D Simulation Study. *IEEE Trans. Electron Devices* , 2009.
[30] X. Fu et al. NBTI tolerant microarchitecture design in the presence of process variation. In *MICRO*, 2008.
[31] B. Zhao et al. Variation-tolerant non-uniform 3D cache management in die stacked multicore processor. In *MICRO*, 2009.
[32] D. M. Tullsen. Simulation and Modeling of a Simultaneous Multithreading Processor. 1996.
[33] M. Lis et al. Scalable, accurate multicore simulation in the 1000-core era. In *ISPASS*, 2011.
[34] S. Li et al. McPAT: an integrated power, area, and timing modeling framework for multicore and manycore architectures. In *MICRO*, 2009.
[35] N. Muralimanohar et al. CACTI 6.5. HP Laboratories. 2009.
[36] H. Homayoun et al. Dynamically heterogeneous cores through 3D resource pooling. In *HPCA*, 2012.
[37] D. Zhao et al. Temperature Aware Thread Migration in 3D Architecture with Stacked DRAM. In *ISQED*, 2013.
[38] ITRS, 2009. [Online]. Available: http://www.itrs.net.
[39]L. Zhang et al. Scheduled Voltage Scaling for Increasing Lifetime in the Presence of NBTI. In *ASP-DAC*, 2009.
[40] H. Tajik et al. Policies for Variation Aware Wearout Management in 3D Multicore Architecture using VAWOM, CECS Technical Report 13-01, UC Irvine, 2013.

9. Appendix

9.1 Incorporating Temperature and mode change in VAWOM

In VAWOM, two key parameters determine the NBTI effect (Vth degradation): the temperature and the operation mode (stress or recovery). VAWOM runs periodically and because of the long period between two consecutive runs, we assume constant temperature between each run and ignore the transient temperature after running VAWOM.

The number of interface traps is calculated at the end of each period. In the next period, with the initial N_{IT}, new temperature, new mode and equations described in [5] [24] we can find the number of interface traps at the end of the period.

Duty cycle is the percentage of time during which a PMOS transistor is under stress. Most papers assume the duty cycle to be 0.5. However, usually the most degraded parts of the circuit lead to timing error and some parts of the circuit may have a duty cycle close to one. Therefore, in this work we assume the duty cycle is equal to one. However, our work is still applicable when using other methods that change the duty cycle of transistors (e.g., [21]).

9.2 Variation Model

Dependence of gate delay to threshold voltage is described in (4) [8]:

$$T_g = \frac{V(1+v_{th}/v_{th}^0)}{(v-v_{th})^\alpha} \quad (4)$$

Where T_g is the delay of the gate, V is the supply voltage, v_{th} is the threshold voltage, v_{th}^0 is the nominal value of v_{th}, and α is typically 1/3.

Each modern core has thousands of critical paths. For a core:

max(T)=probability distribution of longest critical path delay

$f = 1/\max(T)$

Where T is the critical path delay and f is the frequency. We use this model to measure the frequency of different cores and explore the effects of NBTI induced threshold voltage shift on the frequency of the cores.

9.3 VAWOM's Temperature-Aware scheme

Figure 6 shows the activity migration policy for VAWOM's *aging-balancing* algorithm. Array T contains core temperatures and array D contains current degradation of each core. *c_h* is a hash table with current degradation as its key and core number as its value. *t_h* is a hash table with temperature as its key and task number as its value.

```
Inputs: CoreTemp: T[1..n]; CoreDegradation: D[1..n];
CoreHash: c_h[degradation]→{1,...,n}; TempHash:
t_h[Temp]→{1,...,n}
Sort_ascending(T)
Sort_descending(D)
For i = 1 to n:   ---<new task assignment>
    Task[ t_h(T[i])] → Core[c_h(D[i])]
Run (benchmark)
D ←D + new_degradation    ---<update degradation>
```

Figure 6. Aging-balancing activity migration algorithm in VAWOM's temperature-aware scheme.

We list several variations of VAWOM's temperature-aware policies, that combine the core layer migration policies, and the cache layer proactive recovery policies described earlier. In the paper, we evaluate the efficacy of these policies for extending the lifetime reliability of the chip compared to our baseline architecture (*No Change* policy in Table 1) where there is no recovery support.

Table 1- VAWOM temperature-aware policies.

VAWOM temperature-aware policies	Policy in the Core Layer	Policy in the Cache Layer
No Change (baseline)	no change	no change
No Change - RR	no change	recovery in round robin manner
T-Balancing - RR	temperature-balancing algorithm	recovery in round robin manner
A-balancing -Max	aging-balancing algorithm	recover max degraded cache bank

9.4 VAWOM's Variation-Aware scheme

Figure 7 describes two basic policies for VAWOM's process variation-aware scheme. *T*, *co_init*, *co_d*, *ca_init*, and *ca_d* are 5 arrays containing the temperature of cores, core initial threshold voltage, core degradation, cache initial threshold voltage, and cache degradation respectively. *ca_hash* and *co_hash* map total threshold voltage to caches and cores respectively. *task_hash* maps temperatures to the tasks. In the core layer based policy, after sorting the temperature and sorting the current speed of the cores, colder tasks will be assigned to the slowest cores. In the cache layer based policy, after sorting the core temperatures and sorting the current speed of the cache banks, colder tasks will be assigned to the cores adjacent to the slower cache banks.

```
Inputs: CoreTemp: T[1..n]; CoreInitThresh: co_init[1..n];
CoreDegrade: co_d[1..n]; CacheInitThresh: ca_init[1..n];
CacheDegrade: ca_d[1..n]; Cachehash: ca_hash[ca_init +
ca_d]→{1,...,n}; CoreHash: co_hash[co_init + co_d]→{1,...,n};
TaskHash: task_hash[T]→{1,...,n}
1- Core Layer Based:
For i = 1 to n:    ---<calculate core speed>
    A[i] = co_init[i] + co_d[i]
Sort_ascending(T)
Sort_descending(A)
For i = 1 to n:    ---<task assignment>
    Task[task_hash(T[i])] → Core[co_hash(A[i])]
Run (benchmark)
T ← new Temperature
co_d ←co_d + new_degradation    ---<update degradation>
2- Cache Layer Based:
For i = 1 to n:    ---<calculate cache speed>
    A[i] = ca_init[i] +ca_d[i]
Sort_ascending(T)
Sort_descending(A)
For i = 1 to n:    ---<task assignment>
    Task[task_hash(T[i])] → Core[ca_hash(A[i])]
Run (benchmark)
T ← new Temperature
ca_d ←ca_d + new_degradation    ---<update degradation>
```

Figure 7. Migration policies for VAWOM's process variation-aware scheme.

Similar to the temperature-aware scheme, VAWOM's process variation-aware scheme uses one of the *round-robin* or *most degraded* policies for cache layer proactive recovery (Table 2). In the paper, we examine a combination of various migration and proactive recovery policies for this process variation-aware scheme. Maximum threshold voltage shift with respect to nominal voltage is our metric to measure the efficacy of each of these policies.

Table 2- VAWOM process variation-aware policies.

VAWOM process variation-aware policies	Policy in the Core Layer	Policy in the Cache Layer
No Change (baseline)	no change	no change
Core Based	migration of hotter tasks to faster cores	no change
Cache Based	migration of hotter tasks to cores with adjacent faster bank	no change
Interleaved	Periodically select between core based and cache based	no change
Interleaved+RR	Periodically select between core based and cache based	recovery in round robin manner
Interleaved+Max	Periodically select between core based and cache based	recover max degraded cache bank

9.5 Experimental Setup and Parameters

In each VAWOM scheme, a number of cache memory banks and their associated routers need to be disabled. By disabling routers, change of routing is inevitable. We used x-y routing as the base routing algorithm. For deadlock-free routing in presence of disabled routers we used the method proposed in [27].

We examined different periods for running VAWOM, from 5 to 10 seconds and the results didn't show a significant difference. We picked 10 seconds to be the period of VAWOM. This number keeps the overheads low. In addition, larger period provides enough time to reach the stable temperature which is more consistent with the constant temperature assumption between two consecutive runs. Based on these results, any number between 5 and 10 seconds can be used as the period of VAWOM. This frequency of activity migration may not reduce the peak temperature, but will reduce the average temperature of cache banks over time.

Table 3 summarizes the configuration parameters used in our study. Each application in the workloads is fast-forwarded by 200 million instructions to skip part of the initialization phase and then simulated until each thread executes 200 million instructions.

Table 3- System Configuration

Processor Cores	16 dual issue cores (out-of-order), 2 GHz
L1 Instruction/Data Cache	8KB, 4-way (LRU), 64 B Blocks, 2 cycle
L2 cache (shared)	8 MB NUCA, 16 banks, 8-way (LRU), 64 B Blocks, 20 cycle latency (nominal per bank)
Memory Latency	250 cycles
Interconnect	NoC of mesh (4x4 banks and cores) 32-byte links (2 flits per memory access), 1-cycle link latency, 2-cycle router, XY routing