INTEGRATION, the VLSI journal 51 (2015) 10–20

Contents lists available at ScienceDirect

INTEGRATION, the VLSI journal

journal homepage: www.elsevier.com/locate/vlsi

NBTI alleviation on FinFET-made GPUs by utilizing device heterogeneity

Ying Zhang*, Sui Chen, Lu Peng, Shaoming Chen

Division of Electrical & Computer Engineering, School of Electrical Engineering and Computer Science, Louisiana State University, Baton Rouge, LA 70803, United States

ARTICLE INFO

Article history:
Received 10 December 2014
Received in revised form
26 March 2015
Accepted 15 April 2015
Available online 25 April 2015

Keywords:
NBTI
FinFET
Reliability
Heterogeneity

ABSTRACT

Recent experimental studies reveal that FinFET devices commercialized in recent years tend to suffer from more severe NBTI degradation compared to planar transistors, necessitating effective techniques on processors built with FinFET for endurable operations. We propose to address this problem by exploiting the device heterogeneity and leveraging the slower NBTI aging rate manifested on the planar devices. We focus on modern graphics processing units in this study due to their wide usage in the current community. We validate the effectiveness of the technique by applying it to the warp scheduler and L2 cache, and demonstrate that NBTI degradation is considerably alleviated with slight performance overhead.

1. Introduction

As we shift into the deep submicron era, innovative materials and device architectures are becoming ever demanding to continue the trend toward smaller and faster transistors. Among all candidates in investigation, the fin field-effect-transistor (FinFET) stands as one of the most promising substitutes for traditional devices at the ensuing technology nodes, since it presents several key advantages over its planar counterpart [1–4]. By wrapping the conducting channel with a thin vertical "fin" which forms the body of the device, the gate is coupled tighter with the channel, increasing the surface area of the gate-channel interface and allowing much stronger control over the conducting channel [1]. this effectively relieve the so-called short channel effects (SCE) that are observed on planar transistors manufactured with sub-32 nm technology, which in turn implies that FinFET device can provide superior scalability in the deep submicron regime [1].

Another cornerstone motivating the realization of FinFET is the potential performance gain. FinFET transistors can be designed with lower threshold voltage (V_t) and operate with higher drive current, leading to faster switching speed compared to conventional planar devices [1]. Released documents from industry demonstrate that the FinFET transistor persistently demonstrates shorter delay than the planar one while the support voltage is varying, enabling the design and manufacturing of faster processors. Public documents from leading manufacturers also show that the FinFET structure is capable of largely decreasing leakage when the transistor is off [1]. Recently, the Ivy Bridge [5] and Haswell central processing units [6] released by Intel have commercialized this structure (i.e., referred to as "Tri-gate transistor" by Intel), which is also expected to be adopted by other semiconductor manufacturers on their upcoming products [7].

Nonetheless, FinFET is not an impeccable replacement of traditional devices as it raises many challenges to the current industry. One of the most daunting conundrums is the increasing aging rate caused by negative bias temperature instability (NBTI). Recent experimental studies demonstrate that FinFET transistors are more vulnerable to NBTI, leading to a shorter lifetime than a planar device [8,9]. The NBTI aging rate is evaluated by the increase of delay on the critical path after a certain amount of service time. A chip is considered as failed when the delay increment exceeds a pre-defined value after which the timing logic of the processor cannot function correctly. Under the same operation condition, the FinFET device is observed to degrade much faster than the planar counterpart, implying a significantly reduced service lifespan of the target processor. This clearly spurs the development of new techniques to circumvent this problem and prolong the lifetime of FinFET-made processors.

Fortunately, a brief comparison between the main features of FinFET and planar devices sheds some light on alleviating the NBTI effect on future processors. By effectively exploiting the device heterogeneity and leveraging the higher NBTI immunity of planar transistors, the aging of the FinFET structures can be largely suppressed. In this paper, we propose a technique built on top of this principle to improve the durability of FinFET processors.

* Corresponding author.
E-mail addresses: ying.esz.zhang@gmail.com (Y. Zhang), csui1@lsu.edu (S. Chen), lpeng@lsu.edu (L. Peng), schen26@lsu.edu (S. Chen).

http://dx.doi.org/10.1016/j.vlsi.2015.04.003
0167-9260/

In general, our technique is implemented by replacing an existing structure with a planar-device equivalent. Along with minor modifications at the architectural level, our proposed technique is essentially transferring the "aging stress" from the vulnerable FinFET components to the more NBTI-tolerable planar structures, which in turn lower down the temperature on the structure in study, and thus considerably mitigate the NBTI degradation. Note that the proposed scheme is practically feasible because of the good compatibility between the FinFET and planar process technology [10–12].

Considering that the general-purpose graphics processing unit is becoming an increasingly important component in a wide spectrum of computing platforms, we choose a modern GPU as the target architecture to evaluate the effectiveness of our proposed strategy. In this paper, we mainly concentrate on optimizing the reliability of the warp scheduler because of its importance. However, the technique described in this paper can be simply applied to CPU for NBTI mitigation as well. In general, the main contributions of this work are as follows:

- We propose a hybrid-device warp scheduler for reliable operation. By decoupling the warp scheduling into two steps of operations and conducting the prerequisites evaluation in a planar-device structure, we eliminate a large amount of read accesses to the FinFET scheduler hardware and considerably alleviate the NBTI effect.
- We develop a hybrid-device sequential-access cache architecture. All memory requests to this cache hierarchy are handled in a serialized fashion that the tag-array made of planar transistors is probed first and the matching block in the FinFET data array is only accessed on a cache hit. This significantly reduce the activity on the cache data array and improve its reliability.

2. Background

2.1. NBTI degradation mechanism

Negative bias temperature instability is becoming one of dominant reliability concerns for nanoscale P-MOSFETs. It is caused by the interaction of silicon–hydrogen (Si–H) and the inversion charge at the Si/oxide interface [13,14]. When a negative voltage is applied at the gate of PMOS transistors, the Si–H bonds are progressively dissociated and H atoms diffuse into the gate oxide. This process eventually breaks the interface between the gate oxide and the conducting channel, leaving positive traps behind. As a consequence, the threshold voltage of the PMOS transistor is increased, which in turn elongates the switching delay of the device through the alpha power law [15]:

$$T_s \propto \frac{V_{dd}L_{eff}}{\mu(V_{dd}-V_t)^{\alpha}} \tag{1}$$

where μ is the mobility of carriers, α is the velocity saturation index and approximates to 1.3. L_{eff} denotes the channel length. The process described above is termed the "stress" phase where the threshold voltage is persistently increasing with the service time, modeled by the following equation [9].

$$\Delta V_{tstress} = \left(\frac{qT_{ox}}{E_{ox}}\right)^{1.5} \cdot K \cdot \sqrt{C_{ox}(V_{gs}-V_t)} \cdot e^{\frac{-E_a}{4kT}+\frac{2(V_{gs}-V_t)}{T_{ox}E_{01}}} \cdot T_0^{-0.25} \cdot T_{stress} \tag{2}$$

However, when the stress voltage is removed from the gate, H atoms in the traps can diffuse back to the interface and repair the broken bond. This results in a decrease in the threshold voltage, thus termed the "recovery" stage. This iterative stress-recovery processes lead to a saw-tooth variation of the threshold voltage throughout the device's lifespan. The final V_t increase taking both stress and recovery into account can be computed as:

$$\Delta V_t = \Delta V_{tstress} \cdot \left(1 - \frac{2\xi_1 T_{ox} + \sqrt{\xi_2 e^{\frac{-E_a}{kT}} T_0 T_{stress}}}{(1+\delta)T_{ox} + \sqrt{e^{\frac{-E_a}{kT}}(T_{stress}+T_{recovery})}}\right) \tag{3}$$

Note that in Eqs. (2) and (3), T_{stress} and $T_{recovery}$ respectively denote the time under stress and recovery. Other parameters are either constants or material-dependent variables and are listed in Section 4.

That FinFET devices are more vulnerable to NBTI is generally attributed to its unique non-planar architecture, which is visualized by Fig. 1. As can be seen, compared to a traditional planar transistor, the FinFET structure is designed with additional fin sidewall surface with higher availability of Si–H bonds [8,9], implying larger chances of forming interface trap and consequently expediting the device degradation.

The NBTI aging rate depends on multiple factors including both circuit parameters and workload execution patterns. In general, it is acknowledged that voltage, temperature, and the stress/recovery time have strong impact on the aging rate [16,17]. In this work, our proposed techniques significantly reduce the accesses to the target structures, thus lowering down the localized activity and temperature, which is beneficial in enhancing the structure durability.

2.2. Target GPU architecture

The prevalence of unified programming language (e.g., CUDA and OpenCL) has made the general-purpose graphics processing unit a core component in a large variety of systems ranging from personal computers to high-performance computing clusters. Therefore, it is highly important to alleviate the NBTI degradation on this ever increasingly important platform.

Fig. 2 shows the architectural organization of a representative GPU. Note that we follow the Nvidia terminology to depict the processor architecture. As can be seen, the major component of a modern GPU is an array of Streaming Multiprocessors (SMs), each of which contains an amount of CUDA cores (SPs), load/store units and special function units (SFUs). A CUDA core is responsible for performing integer ALU and floating point operations while the

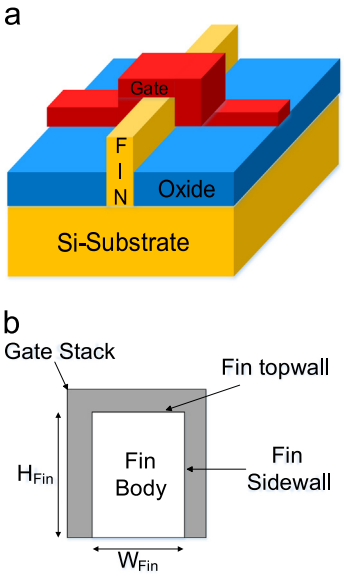


Fig. 1. FinFET transistor structure: (a) overview (b) side view.

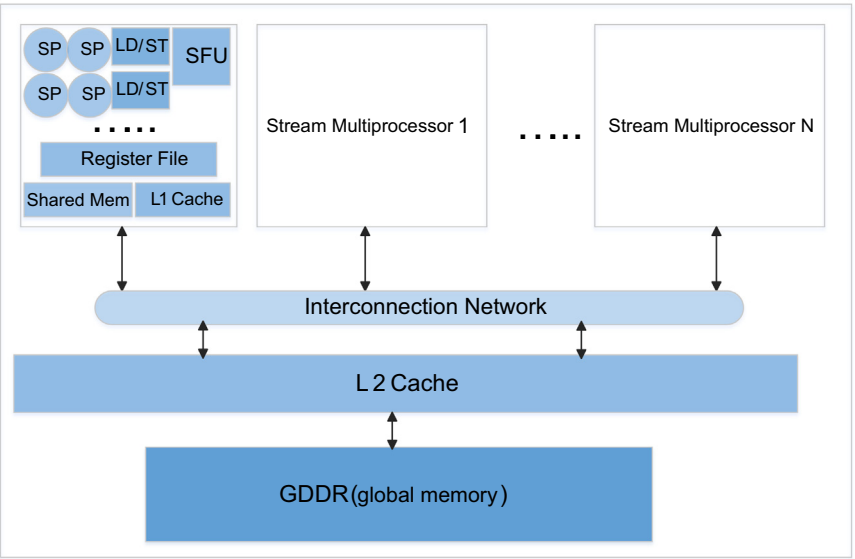


Fig. 2. An illustration of typical GPGPU architecture .

SFUs are devoted to conducting transcendental operations such as sine, cosine, and square root. Each stream multiprocessor also contains a register file, a shared memory and a level 1 cache (usually including instruction/data/constant/texture caches) that are shared among all threads assigned to the SM. All stream multiprocessors connect to an interconnection network, which transfers the memory requests/services between the SMs and the shared L2 cache.

An application developed in CUDA (or OpenCL) contains at least one kernel running on the GPU. A typical kernel includes several blocks composed of substantial threads. During a kernel execution, multiple blocks are assigned to an SM according to the resource requirement. A group of threads from the same block form a warp treated as the smallest scheduling unit to be run on the hardware function units in an SIMT fashion.

3. Hybrid-device warp scheduler

As an emerging platform targeting for massively parallel computing domains, a modern GPU is designed with several unique characteristics different from a regular CPU. In this section, we concentrate on the warp scheduler because it is an important structure that is frequently accessed during program execution. By observing representative execution behaviors of a large collection of GPU applications, we propose a technique exploiting the device heterogeneity to alleviate the NBTI degradation. As we will demonstrate shortly, the proposed technique does not introduce any additional component to the existing GPU architecture, thus minimizing the hardware cost for the implementation.

3.1. Opportunity for improvement

To improve the thread-level parallelism (TLP) and maximize the execution throughput, a modern GPU usually allows multiple warps to reside on the same streaming multiprocessor and hide the execution latencies by switching among those resident warps. At any instant, a warp is considered as ready for execution only when several constraints are simultaneously satisfied.

A first-order prerequisite is the functional correctness, which is secured by ensuring data dependencies between warp instructions. When a warp cannot be dispatched because of unsatisfied data dependency, it should wait until all of its operands are ready. A scoreboard hardware structure is responsible for keeping track of data dependencies in a modern GPU. In addition, warps on a streaming multiprocessor contend for limited functional units. When the dispatch port of the functional unit that a warp needs to use is not vacant, the warp cannot be issued even when its data dependencies have been satisfied.

The warp scheduler is an SRAM hardware structure in charge of selecting candidates from all resident warps to dispatch. For the purpose of high performance, a warp scheduler is capable of dispatching one warp per clock cycle, requiring that scanning through all the scoreboard entries and querying the dispatch ports of all functional units should be performed at each cycle [18,19]. Fig. 3 shows the high-level organization of a warp scheduler equipped in an SM to elaborate the scheduling process. As shown in the figure, all entries, each of which stores complete information of a warp instruction, are going through the conditions checking in parallel in order to identify the candidates ready for execution. Note that to minimize the delay, the scheduler must read the detailed information of a warp (warp ID, opcode, etc.) while evaluating the constraints so that it can dispatch warps as soon as they are ready. Selected warps are sent to the appropriate function units according to the instruction opcode afterwards.

This particular design naturally inspires a technique to mitigate the NBTI degradation on the scheduler. If the readiness of all warp instructions are known ahead via a certain “predicate”, then only the entries with all constraints met are accessed, which in turn decrease the localized activity and temperature, and improve the structure durability.

To justify the potential effectiveness of this strategy, we run a wide spectrum of GPU applications, aiming to observe typical behaviors on the warp scheduler. Fig. 4 shows a snapshot of the warp scheduler's behavior when *WP* is running on a GPU in order to exemplify the activity on the scheduler. The horizontal axis corresponds to the elapsed time and the vertical axis represents the accumulative number of ready warps at each time interval. The number is collected every 50 cycles. With this setting, the maximum number of ready warps cannot exceed 100 on each sampling point considering that two warp instructions can be issued at each cycle. As can be seen from the figure, there are a large amount of execution periods with number of ready warps far less than the theoretical peak, implying a significant reduction in accesses to the scheduler entries in potential. We generally observe that, at any given instant, less than 35% of all the warps

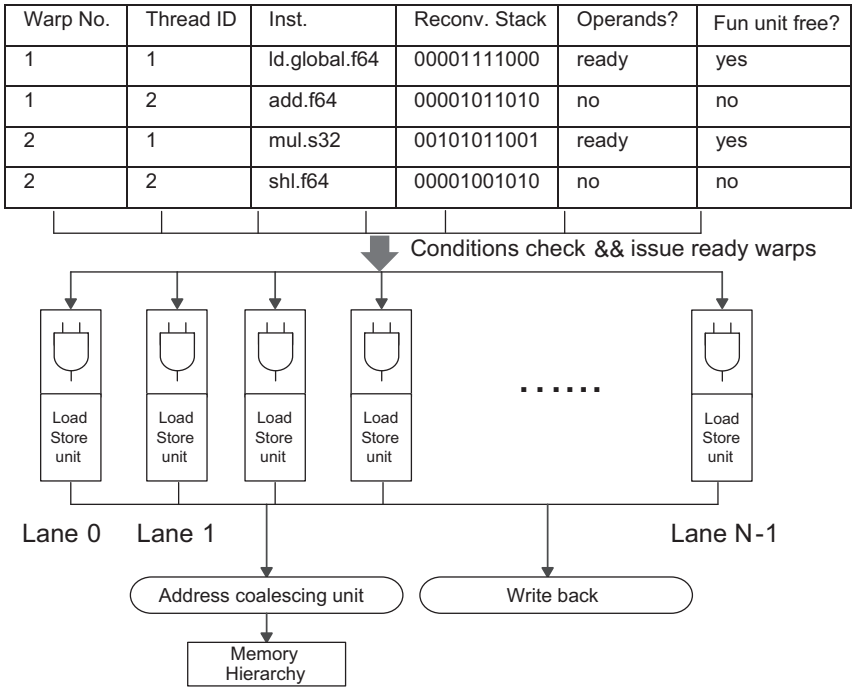


Fig. 3. The architecture of the warp scheduler.

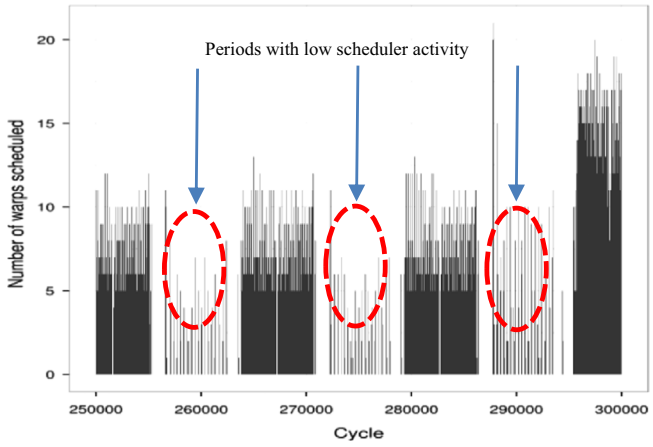


Fig. 4. A snapshot of the scheduler activity while running *WP* Periods with low scheduler activity.

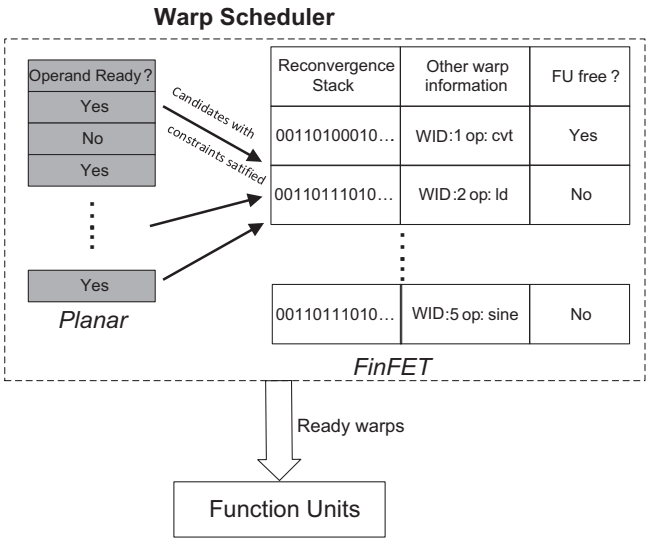


Fig. 5. The architecture of hybrid-device 2-stage scheduler.

have the two prerequisites satisfied for all the tested benchmarks. This is not surprising given the large number of resident warps that a modern GPU spawns for a representative application. As a consequence, the scheduler needs to sweep substantial entries corresponding to the resident warps in order to maximize the opportunity of finding ready warps, which implies large headroom for the reliability optimization.

3.2. *Two-stage scheduling*

Our proposed technique to enhance the durability of the warp scheduler stems from the aforementioned fact at the first place. In order to identify the ready warps, the baseline scheduler is decoupled into two components as shown in Fig. 5. By doing so, the prerequisites checking is extracted from the original parallel accesses and is performed prior to obtaining the detailed information of warp instructions. This checking operation outputs the ID of all available candidates resided on the SM, triggering the consequent accesses to the hardware structure which stores all necessary information to dispatch ready warps based on the specific scheduling policy. If a large amount of resident warps are eliminated from the candidate list due to the violation of scheduling constraints, substantial accesses to the scheduler hardware (i.e., the structure at the right side in Fig. 5) can be avoided.

A non-trivial issue requiring careful consideration in this particular scheduler design is what information should be checked in the first stage. Theoretically, evaluating more scheduling prerequisites would filter larger number of accesses since only the common set of candidates that satisfy each individual constraints are allowed to continue the second stage. However, for certain conditions, checking them in the first stage would lead to undesirable execution behavior because their evaluation results might be changed in the following cycle. The checking on function units' (FU) availability falls into this category. This is because that

the FU status is updated every cycle and a function unit that appears to be free in the current cycle is not necessarily available in the following cycle, if it is assigned to another warp instruction. Therefore in this work, we only check the data dependency in the first stage. As we will demonstrate in section 6, this still results in sufficiently high filter rate for most benchmarks and largely alleviates the NBTI degradation.

On the other hand, considering that the failure of any structure located on the critical path will prevent the entire chip from working correctly, the component where the condition evaluations are conducted tends to become the bottleneck from the perspective of reliability, since all of its entries still need to be scanned every cycle. To overcome this problem, we propose to manufacture this component with the more NBTI-tolerable planar devices. This hybrid-device design effectively leverages the benefits of both devices, aiming to enhance the processor durability. Note that the planar-transistor-made component recording the data dependency and function unit availability is unlikely to suffer from early failure because it only requires one bit for each entry and thus consume negligible power. Also recall that this design is technically feasible due to the good compatibility between FinFET and planar processes as demonstrated in patents [10,12].

Another naturally arising concern with this design is the performance degradation resulted from the sequential scheduler access. Nevertheless, as we will demonstrate in section 6, the performances overhead for most applications are fairly small because only actual accesses to the FinFET part of the scheduler introduces an extra cycle delay. In scenarios where none of the resident warps pass the constraints checking, the execution latency is not impacted.

4. Hybrid-device sequential-access L2 cache

It is widely acknowledged by the high performance computation (HPC) community that memory bandwidth is the main bottleneck in a large number of GPU applications. Due to this reason, the shared L2 cache is becoming an increasingly important component on a modern GPU to reduce the contention on the global memory bandwidth [4], implying that improving the reliability of the L2 cache is of great significance to ensure endurable operation of the GPU.

Typically, the L2 cache installed on a contemporary GPU is designed as a set-associative cache with a reasonable size, serving memory requests sent from the stream multiprocessors. To shorten the execution delay, all ways in the tag array and data array of the selected cache set are searched in parallel and if a stored tag equals to the tag in request, the matching cache block from the data array is returned. However, this access procedure is intrinsically unfriendly to reliable operation since it may introduce substantial unnecessary cache accesses in case the requested data block is not present. For example, the application *Blackscholes* demonstrates a close-to-100% miss rate on the L2 cache, meaning that approximately all the memory requests that are missed in the L1 cache need to be transferred to the global memory eventually. In other words, accesses to the L2 cache are completely unnecessary.

Based on this observation, it is straightforward to realize that filtering out the accesses resulting in cache misses is a simple yet effective approach to slow down the NBTI aging on the L2 cache. Since the data array is orders of magnitude larger than the tag array in both area and power consumption, we first concentrate on the optimization of the data array, which is achieved by applying a technique similar to that developed for the warp scheduler. In specific, we serialize the parallel tag/data access into a sequential procedure [20] with which the tag array in the selected cache set is probed first and only in case an matching tag is found, the corresponding block in the data array is accessed. This particular design, as shown in Fig. 6, reduces the accesses to the data array in two-folds, (1) memory requests that results in cache misses (i.e., no matching tag is found) do not generate consequent accesses to the data array, and (2) only the cache block corresponding to the matching tag, instead of all ways in the set, is read to respond the memory request. With this technique, we expect that the accesses to the data array should be considerably reduced, thus the NBTI aging is largely sup-pressed due to the decreasing activity and temperature.

On the other hand, to prevent the tag array from becoming the reliability bottleneck, we exploit the device heterogeneity and propose to build the tag array with planar transistors. As we will show in later sections, this can effectively leverage the planar device's advantage in NBTI-tolerance and guarantee reliable operations on the L2 tag array throughout the expected lifespan. Also note that in the remainder of this paper, we may interchangeably use the terms planar-tag L2, hybrid-device L2, and sequential-access L2 to refer to this design.

5. Experimental setup

We validate the proposed techniques using a modified GPGPU-Sim 3.1 [21], a cycle-accurate GPGPU simulator. GPUWattch [22] and HotSpot 5.0 [23] are integrated in the simulator for power and temperature calculation, respectively. The chip floorplan required by HotSpot is calibrated against the one used in a recent paper focusing on GPU thermal management [24]. The target architecture is configured based on a Fermi GTX 480 [25] that is widely used in many high-performance computers. Table 1 shows the detailed architectural parameters for our simulation.

The reliability degradation caused by NBTI is usually derived by applying a particular stress pattern to the structures for a time period [14,32]. We follow this approach in this work. Specifically, we choose a set of programs from several benchmark suites [21,26,27], representing typical HPC applications derived from different domains. A full list of applications used in this work is shown in Table 2. For each program, we run them till completion and use the execution statistics to mimic distinct workload patterns. To model the NBTI degradation after a 7-year lifespan, we extrapolate the collected activity to represent the load in 7 years under the steady temperature. We report the final increase in the critical path delay as a measurement of the NBTI aging on the hardware. Eqs. (2) and (3) described in section 2.1 are used to compute the variation in the threshold voltage, which in turn translates to the delay increase via Eq. (1). We set the parameters referred by the equations according to recent studies on device

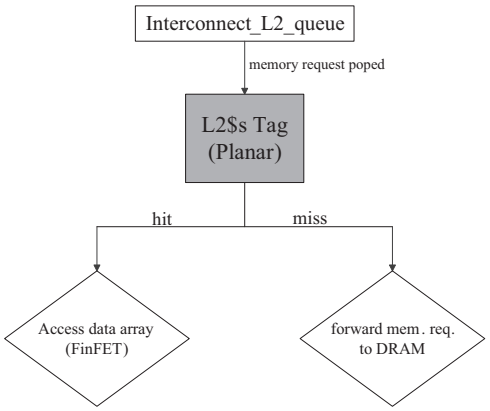


Fig. 6. Workflow of hybrid-device sequential-access L2.

Table 1
Architectural parameters for the GPU in study.

Parameter	Values
#SM	15
#SP	32/SM
LDST units	16/SM
Shared memory	32 kB/SM
L1 data cache	16 kB/SM
Scheduler	Greedy than oldest (GTO)
Core frequency	1400 MHz
Interconnection	1 crossbar/direction
L2 cache	768 kB: 128 cache line size, 16-way associativity. Access latency 5 cycles
L2 frequency	700 MHz
Memory	FR-FCFS scheduling, 64 max. requests/MC
SIMD lane width	16
Threads/warp	32
Technology	22 nm

Table 2
Benchmarks used in this work.

#	Application	Domains
1	B+tree	Search
2	Backprop	Pattern recognition
3	Blackscholes	Financial engineering
4	Gaussian	Linear algebra
5	Heartwall	Medical imaging
6	LPS	3D Laplace solver
7	Myocyte	Biological simulation
8	NN	Neural network
9	NW	Bioinformatics
10	WP	Weather prediction

Table 3
parameter values for computing nbti.

Parameters	FinFET value	Planar value	Description
T_{ox}	1.2 nm	1 nm	Effective oxide thickness
V_t	0.179 V	0.3 V	Threshold voltage
E_o	0.335 V/nm	0.12 V/nm	Electrical field
Fixed parameters			
q	1.602×10^{-19}		Electron charge
V_{dd}	0.9 V		Operating voltage
$\mathcal{E}_{ox}$	1.26×10^{-19} F/m		Permittivity of gate oxide
ξ_1	0.9		Other constants
ξ_2	0.5		
k	8.6174×10^{-5} eV/K		
δ	0.5		
T0	10^{-8} s/nm^2		

features [9,23,28]. Table 3 shows the specific parameter values used in this paper.

6. Result analysis

6.1. Warp scheduler

6.1.1. Improvement on reliability

Fig. 7 shows the NBTI degradation in terms of the increase in scheduler delay on both the baseline GPU and the one with hybrid-device 2-stage warp scheduler. Note that in the figure, the bars marked by "2-stage" refer to the proposed design. A higher delay increase indicates more severe NBTI degradation. As

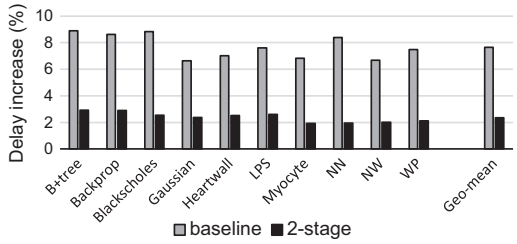


Fig. 7. The NBTI degradation on the warp scheduler.

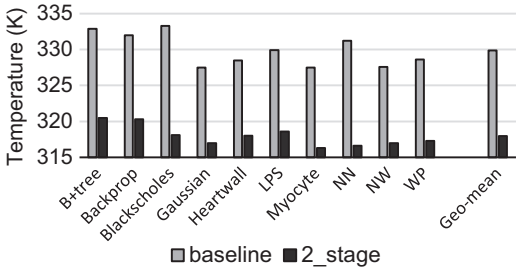


Fig. 8. The steady temperature on the warp scheduler .

can be observed, the aging due to NBTI on the scheduler hardware is largely suppressed for all benchmarks under investigation when the proposed technique is applied. On average, the hybrid-device 2-stage scheduler presents merely 2.36% longer delay after the designed service life, reduced from 7.7% on the baseline GPU.

While the general improvement on the durability is significant, however, it is notable that the benefits corresponding to different workloads are obviously distinct. For example, the load represented by *NN* causes the scheduler delay to be prolonged by around 8.4% after 7 years services on the baseline GPU. With the adoption of the proposed technique, this degradation can be reduced to 1.96%. On the other hand, an execution pattern similar to *Backprop* prevents the scheduler obtaining the same amount of benefit from the technique. Specifically, the scheduler still suffers from 2.9% longer delay after employing the hybrid-device design, while the baseline platform shows 8.6% longer delay that is similar to the degradation corresponding to *NN*.

Considering the exponential relationship between temperature and NBTI degradation, we collect the localized temperature on the scheduler hardware and shows it in Fig. 8 for further analysis. Not surprisingly,

although the proposed technique can significantly cool down the scheduler in most cases, we note that the temperature reductions are apparently different among the evaluated programs, which is similar to the observation made from Fig. 7. When executing *NN*, the temperature on the scheduler is reduced by up to 15 °C, whereas the temperature reduction for *Backprop* is about 11 °C. To gain more insights into the reason behind this phenomenon, let us recall the rationale of the 2-stage scheduler that is described in section 3.2. The essential reason for the reduced scheduler accesses is that a large amount of prerequisite evaluations turn out to be false, thus the unnecessary operations on the "unready warps" are avoided. In other words, how much benefit can be obtained from the proposed technique largely depends on the amount of accesses that can be filtered. Table 4 shows the percentage of accesses saved by the constraint checking stage. As can be seen, the data dependency checking stage can generally filter out more than 92% of accesses to the scheduler, thus considerably enhancing the durability of the hardware. In particular, we note that 76.9% of scheduler accesses when executing *Backprop* are dispensable, while for *NN* this ratio rises up to 97.4%, implying higher possibilities to lower the power and temperature on the scheduler.

We also plot the power consumption of the scheduler in Fig. 9 to visualize the changes on the scheduler activity. Clearly, the hybrid-device 2-stage scheduler significantly reduces the scheduler power for all evaluated benchmarks, which in turn lowers the local temperature and improves the hardware durability.

6.1.2. Performance overhead

The extra cycle introduced by the 2-stage scheduler is likely to result in undesirable performance overhead for the program execution. Fig. 10 shows the performance in terms of normalized IPC (normalized to the baseline GPU) of all benchmarks running on a GPU with the 2-stage scheduler. It is straightforward to note that the performance degradation is distinct among the program collection. In this subsection, we briefly analyze the possible impact on the performance due to the extra cycle and explain the different performance degradations.

The GPU's massive parallelism may be able to hide part of the extra latency during the execution depending on the features of

Table 4
Filter rate on the first stage of warp scheduler.

Application	Filter Rate
B+tree	75.82%
Backprop	76.93%
Blackscholes	88.74%
Gaussian	98.82%
Heartwall	88.46%
LPS	90.59%
Myocyte	99.85%
NN	97.41%
NW	97.70%
WP	99.49%

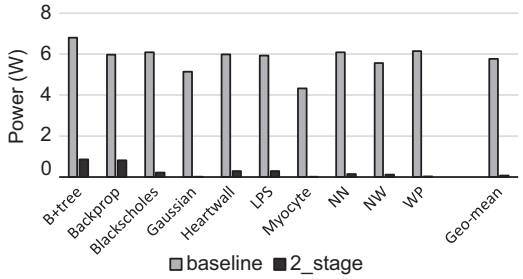


Fig. 9. The power consumed by the warp scheduler.

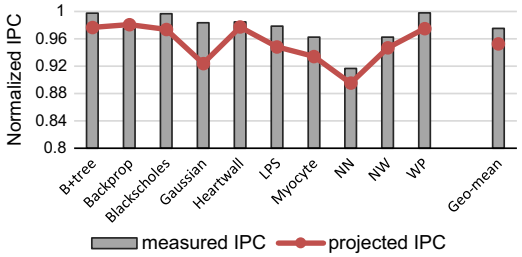


Fig. 10. Normalized IPC on the GPU with 2-stage scheduler.

applications. We use the terms "longest warp" and "longest-warp chain" to explain the latency manifested in the results. We define "longest warp" as the warp with the longest running time during a kernel launch and "longest-warp-chain" as the set of longest warps in each of the sequence of kernel launches in the lifetime of an application. In a typical GPU application, the running time of a longest-warp chain is the sum of execution latencies of all warps in the chain because (a) when a kernel is launched, all its warps are started simultaneously and (b) a kernel is not launched until all warps of the previous kernel launch complete. In other words, latency on the longest warp could not be hidden as easily as that on other warps. Longest warps also do not overlap temporally. For each longest warp we can compute its average latency as:

$$AvgLatency = \frac{\sum_{n=1}^{N} C_n}{\sum_{n=1}^{N} I_n} \tag{4}$$

where N is the number of kernel launches and C_n and I_n are respectively the number of cycles and warp instructions of the longest warp in each kernel launch.

The I_n instructions in a kernel launch are the instructions issued to and executed by a warp. The extra cycle introduced to the scheduler will be added before each of the instructions is executed. Since the instructions are executed in-order, this is equivalent to adding $\sum_{n=1}^{N} I_n$ extra cycles to the entire longest-warp chain. The average latency of the warp after adding the extra cycles should become:

$$AvgLatency_{delayed} = \frac{\sum_{n=1}^{N} C_n + \sum_{n=1}^{N} I_n}{\sum_{n=1}^{N} I_n} \tag{5}$$

The overhead indicators can be deducted from the two latencies shown below:

$$\begin{aligned} OverheadInd &= \frac{\Delta latency}{AvgLatency} \\ &= \frac{AvgLatency_{delayed} - AvgLatency}{AvgLatency} \\ &= \frac{\frac{\sum_{n=1}^{N} C_n + \sum_{n=1}^{N} I_n}{\sum_{n=1}^{N} I_n} - \frac{\sum_{n=1}^{N} C_n}{\sum_{n=1}^{N} I_n}}{\frac{\sum_{n=1}^{N} C_n}{\sum_{n=1}^{N} I_n}} = \frac{\sum_{i=1}^{N} C_n + \sum_{i=1}^{N} I_n - \sum_{i=1}^{N} C_n}{\sum_{i=1}^{N} C_n} \\ &= \frac{\sum_{i=1}^{N} I_n}{\sum_{i=1}^{N} C_n} = \frac{1}{AvgLatency} \end{aligned} \tag{6}$$

The normalized IPC (measured) and the one derived from the overhead indicator (projected) both are shown in Fig. 10. As the figure shows, they are closely correlated. The average latencies and the overheads are determined by the behaviors of the longest warps which are in turn closely related to the characteristics of individual applications. For example, *B+tree* involves a kernel launch with 48 warps on each SM and initiates many global memory transactions (159.26 per cycle). Its longest warp has an average delay of more than 100 cycles. *NN*, on the other hand, has a much smaller average delay (smaller than 10), because it generates much fewer global memory transactions (only 0.06

per cycle) and each SM executes only 8 warps. With such few memory transactions and fewer warps, each of the warps, including the longest warp, does not have to wait for long-delay memory operations while sharing more computational resources. This different memory request intensities result in average latencies of the longest warp chains as 41.7 and 8.53 cycles for *B+tree* and *NN*, respectively. Consequently, we observe apparently different performance losses for these two benchmarks.

6.2. L2 cache

We now shift our concentration to the L2 cache. For this structure, we first focus on its data array. Fig. 11 shows the NBTI degradation on the L2 cache data array on both the baseline GPU and the GPU with a planar-tag sequential-access L2. Note that the latter one is labeled as "with_Ptag" in the figure, where the capital letter P stands for planar device. As shown in the figure, the general trend is similar to what is observed in previous section that the proposed technique is capable of largely slowing down the aging due to NBTI on the target component throughout the service life. On average, the hybrid-device design reduces the delay increase from 14.1% in the baseline situation to 2.8%.

We also note that the improvement on the durability is different among the programs in study. For example, the applications *Gaussian* and *LPS* causes approximately the same level of NBTI aging on the baseline platform. However, with the hybrid-device L2 cache, running *LPS* apparently leads to less significant NBTI degradation (2.7%) compared to the execution of Gaussian (4.93%). This is resulted from the distinct temperature variations on the L2 while running these programs. Fig. 12 shows the steady L2 temperature for both the baseline and our proposed design. From the figure, we note that on the GPU with the hybrid-device design, running *LPS* makes the L2 cache much cooler compared to the execution of Gaussian. The reason is as follows. Similar to accessing the 2-stage warp scheduler, memory requests sent to the L2 cache are served in a sequential tag-data access pattern, while the tag probing can eliminate the unnecessary accesses to the data array (i.e., cache misses). In other words, the different amount of cache accesses that are avoided are the essential reason for the distinct temperature and reliability changes. Fig. 13(a) and (b) respectively shows the L2 cache miss rates and comparison of L2 power for different applications. As can be seen, *LPS* demonstrates an L2 miss rate of 36%, thus resulting in impressive reduction in L2 power/temperature and great reliability enhancement as a consequence. For *Gaussian*, most of the accesses to the data array cannot be avoided because of the low L2 miss rate (4.7%). This eventually leads to the relatively smaller improvement on the NBTI degradation. Other benchmarks with high L2 miss rates including *Blackscholes* also present relatively larger improvement on device durability compared to those with low L2 miss rates such as *NN*. On the other hand, it is important to keep in mind that even for a cache hit, only the matching block is accessed afterwards. For caches with high associativity, which is the typical design in many modern processors, this provides another fold of reduction in the localized power and temperature. Due to this reason, the power consumption of L2 for all bench-marks is considerably reduced while running with sequential-access cache as shown in Fig. 13(b).

The reliability of the tag array is becoming a major concern in the proposed cache design since the accesses to this structure have not been reduced. Fortunately, due to higher NBTI-immunity manifested by planar transistors and the small power consumed by the tag array, the L2 tag is not likely to suffer from significant NBTI degradation. Fig. 14 compares the NBTI degradation in the tag array with both designs, which is essentially determined by the different NBTI tolerances of FinFET and planar transistors. As can be seen, the tag array made of planar device leads to much less degradation compared to the baseline platform, implying more endurable operation in the service life.

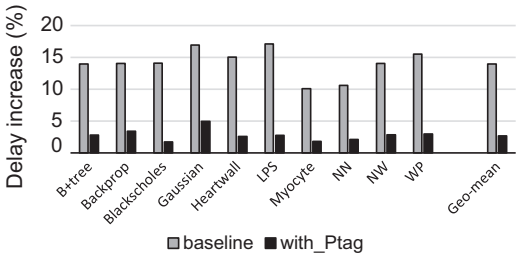


Fig. 11. NBTI degradation on the L2 data array.

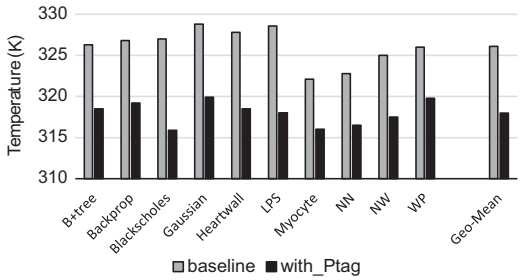


Fig. 12. Steady temperature on the L2 data array.

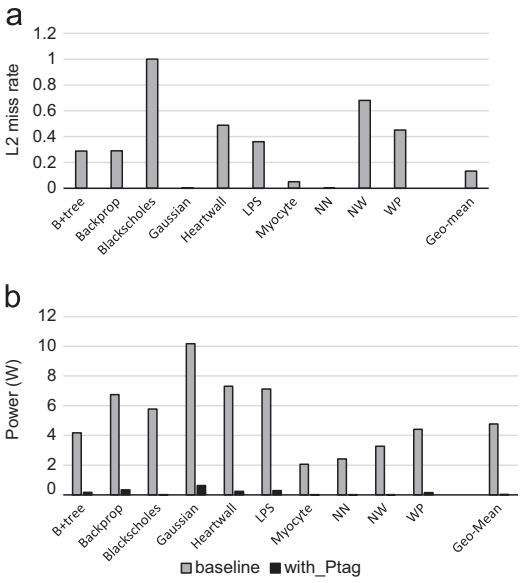


Fig. 13. (a) L2 miss rate, (b) power consumption of the L2 data array.

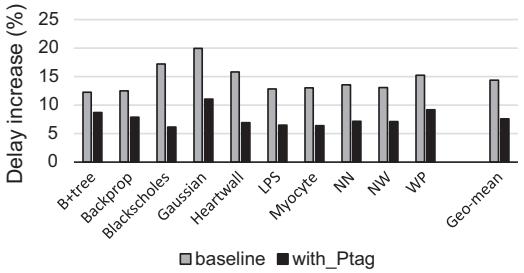


Fig. 14. NBTI degradation on the L2 tag array.

Another concern that deserves evaluation is the possible performance loss resulted from the extra delay spent on the cache tag probing. We demonstrate the normalized IPC of all programs with the sequential-access L2 cache in Fig. 15(a) and find that the performance degradation for all benchmarks in investigation is within 1.5%. This does not go beyond our expectation due to the following reasons. First, only a cache hit introduces an extra cycle delay since misses will be promptly forwarded to the lower memory hierarchy after the tag probing, thus not wasting any cycles. Second, even an L2 cache hit takes multiple cycles to complete. This includes the 5 cycles to access the data array and the time spent on the interconnection network. Therefore, the extra one cycle does not weigh heavily and will not evidently impair the overall performance. Fig. 15(b) shows the average L2 hits per cycle (i.e., actual accesses to the data array) for the program collection in order to briefly explain the different impacts on the performance caused by the extra cycle. As can be observed, applications such as *Blackscholes*, *Myocyte*, and *NW* have extremely low L2 hits intensity, so their performance is not notably degraded (close to zero loss) with the sequential-access L2 cache. On the contrary, Gaussian result in more frequent L2 hits, thus their execution speed is lowered by a relatively higher percentage (1.5%). Nonetheless, based on the evaluations made on the L2 cache, it is still reasonable for us to conclude that the proposed hybrid-device sequential-access design can significantly slow down the NBTI aging on the L2 cache with slight performance overhead.

7. NBTI aging process on duplicated Structures

In addition to utilizing device-level heterogeneity as introduced in prior sections, another common approach to alleviate reliability degradation is to duplicate the vulnerable structures and use them alternatively to slow down the overall aging [33]. In this section, we adopt this technique on the target processor and compare it against our proposed strategy (i.e., mix-device sequential-access) from the NBTI alleviation perspective.

Following this approach, we duplicate the warp scheduler and L2 cache on the target GPU. Each replica stores identical information. During execution, the access to the structure is alternatively made to one of the replicas. By doing so, we aim at reducing the accesses to a single structure and slow down its aging process. However, this approach tends to largely increase the power consumption because of the duplicated structure. Taking this into consideration, we employ the NBTI efficiency metric proposed in a prior work [31] to comprehensively evaluate the effectiveness of this technique. As described in [31], this metric can effectively characterize the efficiency of a NBTI mitigation technique, i.e., whether it can alleviate NBTI degradation with minimum performance/power cost. The definition of NBTI efficiency is given by the following equation. A smaller NBTI efficiency is corresponding to a better technique.

$$NBTI_{efficiency} = (Delay \times (1 + NBTI_{degradation}))^3 \times P$$

where *Delay* indicates performance in terms of execution time, $NBTI_{degradation}$ denotes the transistor switch delay increase due to NBTI aging process while *P* represents the power consumption. In addition, for the parameter $NBTI_{degradation}$, we choose the larger one between the degradation on the warp scheduler and L2 cache, because the NBTI guardband is determined by the largest delay increase.

We run all workloads respectively on three platforms, (1) the baseline processor where all components are made of FinFET, (2) a FinFET processor with duplicated warp scheduler and L2 Cache (i.e., referred as "duplication"), (3) a FinFET GPU with the proposed two-stage mix-device warp scheduler and L2 cache as described in Sections 3 and 4. We report the NBTI efficiency for each workload on these three platforms. The results are shown in Fig. 16. Note that the results are normalized to the baseline processor. As can be seen from the figure, the "duplication" technique does not always result in better NBTI efficiency than the baseline processor. For workloads including *Gaussian* and *LPS*, the NBTI efficiency associated with the structure-duplication technique is obviously worse than that on the baseline processor. This is mainly because the larger power consumption outweighs the benefit obtained from the improved NBTI degradation. Recall the temperature plot shown in Fig. 12. It is easy to notice that *Gaussian* and *LPS* result in noticeably higher temperature than other workloads. Considering the large area occupied by the duplicated L2 cache, this implies significant leakage power increase on the target processor. In contrast, for workloads including *Myocote*, the structure-duplication technique leads to better NBTI efficiency compared to the baseline because of the improvement in NBTI aging. In general, the geometric mean of NBTI efficiency on the "duplication" processor is 1.1% higher than that on the baseline processor. On the other hand, the proposed two-stage mix-device technique outperforms both baseline processor and the processor with duplicated structure. This is not surprising because the sequential-access significantly reduce the dynamic power by filtering out the unnecessary accesses, which also helps reduce the leakage power because of lower temperature. As shown in Fig. 16, the geometric mean of the NBTI efficiency with the proposed technique is 89.1% of the baseline. Therefore, it is

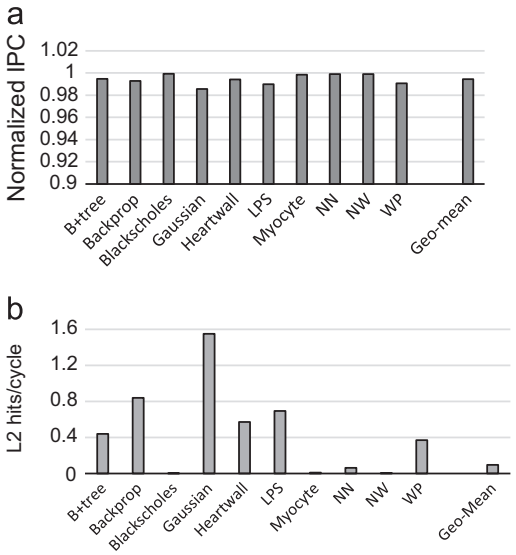


Fig. 15. (a) Normalized IPC with the sequential-access L2 cache, (b) L2 hits/cycle.

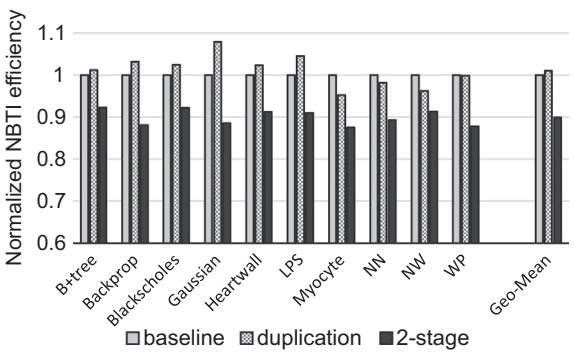


Fig. 16. NBTI efficiency on the three platforms.

reasonable to conclude that our proposed technique can efficiently mitigate the NBTI degradation.

8. Related work

NBTI Mitigation: NBTI has been recognized as a major reliability concern as the semiconductor industry shifts into the deep submicron era. To mitigate the NBTI degradation and enhance the device's durability, researchers have conducted substantial works in the past years. Ramakrishnan et al. [29] introduce an approach to reduce the NBTI wearout in FPGAs by loading the reversing bit patterns in idle periods. Gunadi et al. [13] introduce a scheme called Colt to balance the utilization of devices in a processor for reliability improvement. Specifically focusing on the storage components, Shin et al. [30] propose to proactively set the PMOS transistors to recovery mode, and moving data around free cache arrays during operation.

Converse to these works which attempt to manipulate the time under stress and recovery, Tiwari et al. [14] propose a framework named facelift to combat NBTI degradation by adjusting higher level parameters including operating voltage, threshold voltage and the application scheduling policy. Fu et al. [31] concentrate on the NBTI mitigation in presence of process variation. They effectively utilize the interplay between NBTI aging and process variation to prevent early failure of specific structures.

There are few works aiming to alleviate the NBTI aging on GPUs in literature. Rahimi et al. [32] focus on the GPUs designed in VLIW fashion and present a technique to slow down the NBTI aging for this particular architecture. By exploring the unbalanced usage among function units within a VLIW slot, their proposed strategy can uniformly as-sign the stress among all computation units and achieve an even aging rate.

Characterization of FinFET Reliability: as FinFET is widely considered as an attractive replacement of planar transistors for the next few technology nodes, studies focusing on the reliability of this new structure is becoming fairly important. Lee et al. [34] investigate the NBTI characteristics on SOI and body-tied FinFETs and observe that a narrow fin width leads to more severe degradation than a wider fin width. Crupi et al. [35] compare the reliability of triple-gate and planar FETs. The author show that the behavior of time-dependent dielectric breakdown (TDDB) is not changed on the triple-gate architecture under different gate voltages and temperatures. This is also corroborated in the work conducted by Groeseneken et al. [8], which further demonstrate that FinFET devices tend to suffer from more severe NBTI degradation. In [9], Wang et al. analyze the soft-error resilience of FinFET devices and conclude that FinFET circuit is more reliable than bulk CMOS circuit in terms of soft-error immunity.

Hybrid-device Design: exploiting device-level heterogeneity have been widely used for performance and energy efficiency optimization in computer architecture study. Saripalli et al. [36,37] discuss the feasibility of technology-heterogeneous cores and demonstrate the design of mix-device memory. Wu et al. [38] presents the advantage of hybrid-device cache. Kultursay [39] and Swaminathan [40] respectively introduce a few runtime schemes to improve performance and energy efficiency on CMOS–TFET hybrid CMPs. For the optimization on GPUs, Goswami et al. [41] propose to integrate resistive memory into the compute core for reducing the power consumption on GPU register file.

Our work deviates from the aforementioned studies in that we aim to alleviating the NBTI degradation on GPUs made of FinFET from the architectural level. In addition, compared to our prior work [42], this study extends the application of the proposed technique to more structures and thus provides more general guidance to the processor design.

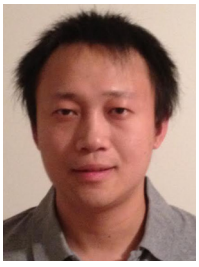
9. Conclusion

FinFET technology is recognized as a promising substitute of conventional planar devices for building processors in the next decade due to its better scalability. However, recent experimental studies demonstrate that FinFET tends to suffer from more severe NBTI degradation compared to the planar counterpart. In this work, we focus on the NBTI reliability issue of a modern GPU made of FinFET and propose to address this problem by exploiting the device heterogeneity. We introduce a set of techniques that merely involve minor modifications to the existing GPU architectures. The proposed techniques leverage planar devices' higher immunity to NBTI and are effective in slowing down the aging rate of the device. Our evaluation results demonstrate that the minor changes to the warp scheduler and the L2 cache can considerably alleviate the degradation due to NBTI with slight performance overhead.

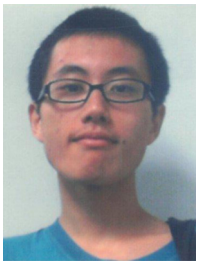
References

[1] Intel Corporation. Intel's revolutionary 22 nm transistor technology, May 2011.
[2] A. Asenov, C. Alexander, C. Riddet, and E. Towie. Predicting future technology performance, in: Proceedings of 50th Design Automation Conference (DAC), Jun. 2013.
[3] A. B. Kahng. The ITRS design technology and system drivers roadmap: process and status, in: Proceedings of the 50th Design Automation Conference (DAC), Jun. 2013.
[4] V. B. Kleeberger, H. Graeb, and U. Schlichtmann, Predicting future product performance: modeling and evaluation of standard cells in FinFET technologies, in: Proceedings of 50th Design Automation Conference (DAC), Jun. 2013.
[5] Intel Corporation, 3rd Generation of Intel Core i7 Processor. ⟨http://ark.intel.com/products/family/65505⟩.
[6] Intel Corporation, 4th Generation of Intel Core i7 Processor. ⟨http://ark.intel.com/products/family/75023⟩.
[7] ⟨http://www.eetimes.com/document.asp?doc_id=1264668⟩.
[8] G. Groeseneken, F. Crupi, A. Shickova, S. Thijs, D. Linten, B. Kaczer, N. Collaert, and M. Jurczak, Reliability issues in MUGFET nanodevices, in: Proceedings of the IEEE 46th Annual International Reliability Physics Symposium (IRPS), April 2008.
[9] Y. Wang, S.D. Cotofana, and L. Fang, Statistical reliability analysis of NBTI impact on FinFET SRAMs and mitigation technique using independent-gate devices, in: Proceedings of the IEEE/ACM International Symposium on Nanoscale Architecture (NANOARCH), Jul. 2012.
[10] B. A. Anderson, A. J. Joseph, and E. J. Nowak, Integrated circuit including FinFET RF switch angled relative to planar MOSFET and related design structure. U.S. Patent 8125007 B2, Feb. 2012.
[11] J.P. Colinge, Multiple-gate SOI MOSFETs, Solid-State Electron. 48 (6) (2004) 897–905.
[12] B. B. Doris, D. C. Boyd, M. Leong, T. S. Kanarsky, J. T. Kedzierski, M. Yang, Hybrid planar and FinFET CMOS devices. U.S. Patent 7250658 B2, Jun. 2007.
[13] E. Gunadi, A. A. Sinkar, N. S. Kim, and M. H. Lipasti, Combating aging with the colt duty cycle equalizer, in Proceedings of 43rd Annual IEEE/ACM International Symposium on Microarchitecture (MICRO), Dec. 2010.
[14] A. Tiwari and J. Torrellas, Facelift: hiding and slowing down aging in multicores, in: Proceedings of the 41st IEEE/ACM International Symposium on Microarchitecture (MICRO), Nov. 2008.
[15] T. Sakurai, R. Newton, Alpha-power law MOSFET model and its applications to CMOS inverter delay and other formulas, IEEE J. Solid-State Circuits (1990).
[16] Predictive Technology Model. ⟨http://ptm.asu.edu⟩.
[17] F. Wang, Y. Xie, K. Bernstein, and Y. Luo, Dependability analysis of nano-scale FinFET circuits, in: Proceedings of the 2006 Emerging VLSI Technologies and Architectures (ISVLSI), Mar. 2006.
[18] J. Hennessy, D. A. Patterson, Computer architecture: a quantitative approach. 5th edition.
[19] H. Kim, R. Vuduc, S. Baghsorkhi, J. Choi, and W. Hu, Performance analysis and tuning for general purpose graphics processing units (GPGPU). http://dx.doi.org/10.2200/S00451ED1V01Y201209CAC020.
[20] Z. Chishti, M. D. Powell, and T. N. Vijaykumar, Distance associativity for high performance energy efficient non-uniform cache architectures. In Proceedings of International Symposium on Microarchitecture (MICRO), Dec. 2003.
[21] A. Bakhoda, G. Yuan, W. Fung, H. Wong, and T. Aamodt, Analyzing CUDA Workloads Using a Detailed GPU Simulator, ISPASS, 2009.
[22] J. Leng, T. Hetherington, A. Eltantawy, S. Gilani, N. S. Kim, T. M. Aamodt, V. J. Reddi. GPUWattch: enabling energy optimizations in GPGPUs, in: Proceedings of International Symposium on Computer Architecture (ISCA), Jun 2013.
[23] Hotspot 5.0 Temperature Modeling Tool.
[24] R. Nath, R. Ayoub, and T. S. Rosing, Temperature aware thread block scheduling in GPGPUs, in: Proceedings of 50th Design Automation Conference (DAC), Jun 2013.
[25] GTX 480 Specifications. ⟨http://www.geforce.com/hardware/desktop-gpus/geforce-gtx-480/specifications⟩.

[26] Nvidia Corporation, CUDA Computing SDK 4.2.
[27] S. Che, M. Boyer, J. Meng, D. Tarjan, J. W. Sheaffer, S.-H. Lee, and K. Skadron, Rodinia: a benchmark suite for heterogeneous computing, in: Proceedings of the IEEE International Symposium on Workload Characterization (IISWC), Oct. 2009.
[28] S. Chaudhuri, and N. K. Jha, 3D vs. 2D analysis of FinFET logic gates under process variations in: Proceedings of 29th International Conference on Computer Design (ICCD), Oct. 2011.
[29] K. Ramakrishnan, S. Suresh, N. Vijaykrishnan, M. J. Irwin, and V. Degalahal, Impact of NBTI on FPGAs, in: Proceedings of the International Conference VLSI Design, Jan. 2007.
[30] J. Shin, V. Zyuban, P. Bose, and T. M. Pinkston, A proactive wearout recovery approach for exploiting microarchitectural redundancy to extend cache SRAM lifetime, in: Proceedings of the International Symposium on Computer Architecture (ISCA), Jun. 2008.
[31] X. Fu, T. Li, and J. Fortes, NBTI tolerant microarchitecture design in the presence of process variations, in: Proceedings of the International Symposium on Microarchitecture (MICRO), Nov. 2008.
[32] A. Rahimi, L. Benini, R. K. Gupta, Aging-aware compiler-directed VLIW assignment for GPGPU architectures, in: Proceedings of 50th Design Automation Conference (DAC), Jun. 2013.
[33] E. Rotenberg, AR-SMT: a microarchitectural approach to fault tolerance in microprocessors, in: Proceedings of the International Symposium on Fault-Tolerant Computing (FTCS), 1999.
[34] H. Lee, C.H. Lee, D. Park, Y-K. Choi., A study of negative-bias temperature instability of SOI and body-tied FinFETs, IEEE Electron Device Lett. 26 (5) (2005) 328.
[35] F. Crupi, B. Kaczer, R. Degraeve, V. Subramanian, P. Srinivasan, E. Simoen, A. Dixit, M. Jurczak, G. Groeseneken, Reliability comparison of triple-gate versus planar SOI FETs, IEEE Trans. Electron Devices 53 (9) (2006) 2351–2357.
[36] V. Saripalli, G. Sun, A. Mishra, Y. Xie, S. Datta, V. Narayanan, Exploiting heterogeneity for energy efficiency in chip multiprocessors, IEEE Trans. Emerg. Sel. Topics Circuits Syst. 1 (2011) 109–119.
[37] V. Saripalli, A. K. Mishra, S. Datta, and V. Narayanan., An energy-efficient heterogeneous CMP based on hybrid TFET-CMOS cores in: Proceedings of the Design Automation Conference, Jun. 2011.
[38] X. Wu, J. Li, L. Zhang, E. Speight, R. Rajamony, and Y. Xie, Hybrid cache architecture with disparate memory technologies, in: Proceedings of the ISCA'09.
[39] E. Kultursay, J, Swaminathan, V. Saripalli, V. Narayanan, M. Kandemir, and S. Datta, Performance enhancement under power con-straints using heterogeneous CMOS-TFET multicores, in: Proceedings of the CODES+ISSS'12.
[40] K. Swaminathan, E. Kultursay, V. Saripalli, V. Narayanan, M. Kandemir, and S. Datta, Improving energy efficiency of multi-threaded applications using heterogeneous CMOS-TFET multi-cores, in: Proceedings of the International Symposium on Low Power Electronics Design (ISLPED)'11.
[41] N. Goswami, B. Cao, and T. Li, Power-performance co-optimization of throughput core architecture using resistive memory, in: Proceedings of the 19th IEEE International Symposium on High Performance Computer Architecture (HPCA), Feb. 2013.
[42] Y. Zhang, S. Chen, L. Peng, and S.-M. Chen, Mitigating NBTI Degradation on FinFET GPUs through Exploiting Device Heterogeneity, in: Proceedings of the IEEE Computer Society Annual Symposium on VLSI (ISVLSI), Tampa, FL, Jul. 2014.



Ying Zhang received the Bachelor's and Master's degree in Electronics and Information Engineering from Huazhong University of Science and Technology, China, in June 2006 and 2008. He received his Ph.D. degree in Electrical and Computer Engineering from Louisiana State University in 2013. He is currently working as a performance architect in Intel Corporation. His research interests include heterogeneous system design and processor reliability. He also has interests in GPU architecture.



Sui Chen received the bachelor's in information security from Shanghai Jiao Tong University, China in 2011. He is currently a Ph.D. student in the Division of Electrical and Computer Engineering, Louisiana State University, Baton Rouge. His research interest lies in numerical resilience.

Lu Peng is currently an Associate Professor with the Division of Electrical and Computer Engineering at Louisiana State University. He received the Bachelor's and Master's degrees in Computer Science and Engineering from Shanghai Jiao Tong University, China. He obtained his Ph.D. degree in Computer Engineering from the University of Florida in Gainesville in April 2005. His research focus on memory hierarchy system, reliability, power efficiency and other issues in CPU design. He also has interests in Network Processors. He received an ORAU Ralph E. Powe Junior Faculty Enhancement Awards in 2007 and a Best Paper Award from IEEE International Conference on Computer Design in 2001. Dr. Peng is a member of the ACM and the IEEE Computer Society.

Shaoming Chen received the bachelor's and master's degrees in Electronic and Information Engineering from Huazhong University of Science and Technology, China. He is currently a Ph.D. student in the Division of Electrical and Computer Engineering at Louisiana State University, majoring in Computer architecture. His research interests cover the cost optimization in data centers and memory hierarchy systems.