Thermal-Aware Clustered Microarchitectures

Pedro Chaparro, José González and Antonio González
Intel Barcelona Research Center – Intel Labs - UPC
{pedro.chaparro.monferrer, pepe.gonzalez, antoniox.gonzalez}@intel.com

Abstract

As frequencies and feature size scale faster than operating voltages, power density is increasing in each processor generation. Power density and the cost of removing the heat it generates are increasing at the same rate. Leakage is significantly increasing every process generation and it is expected to be the main source of power in the near future. Moreover, leakage power grows exponentially with temperature. This paper proposes and evaluates several techniques with two goals: reduction of average temperature in order to decrease leakage power, and reduction of peak temperature in order to reduce cooling cost. Combinations of temperature-aware steering techniques and cluster hopping are investigated in a quad-cluster superscalar microarchitecture. Combining cluster hopping with a temperature-aware steering policy results in 30% reduction in leakage power and 8% reduction in average peak temperature at the expense of a slowdown of just 5%.

1. Introduction

Power dissipation is one of the major hurdles in the design of next-generation microarchitectures. Power density is increasing in each generation due to the fact that feature size and frequency are scaling faster than operating voltage. Power density directly translates into heat, and this heat must be removed from the processor die in order to keep the silicon temperature below a certain limit. In fact, the cost of removing heat is increasing at the same rate as power density. This increase is affecting the processor design in many different ways. For instance, the cooling system of a processor is targeted to support a peak temperature, even though the processor spends most of the time running at much lower temperatures. The cost of the cooling system has been quantified in the order of 1-3$ or more per Watt when the average power exceeds 40 Watts [3][11], which represents an important cost.

Moreover it is expected that within a few process generations the contribution of leakage power to the total power will be comparable to the contribution of dynamic power [3][8]. Leakage power is highly dependent on temperature.

On the other hand, wire delays scale much slower than gate delays [1][22] and will become a serious obstacle to the scalability of superscalar processors. Clustered microarchitectures are an effective paradigm to deal with the problem of wire delays and complexity by means of partitioning some of the processor resources [9], as for instance the processor backend, and attempting to maximize local (and fast) communications and reduce global (and slow) communications.

This paper presents a temperature-aware clustered microarchitecture. We propose several architectural modifications with a twofold objective: first, reducing maximum temperature to cut down the cooling costs, and second, reducing leakage power dissipation by means of controlling average temperature. First, we propose and analyze different thermal-aware heuristics when deciding the destination cluster of a particular instruction (a.k.a. instruction steering). Second, we investigate different cluster hopping with different rotation policies. Finally we evaluate the combination of thermal-aware steering techniques with cluster-hopping.

The rest of the paper is as follows: Section 2 describes the processor architecture and the power and thermal models. Section 3 provides initial temperature, leakage and performance results for the baseline clustered architecture. Section 4 introduces our proposals for the thermal-aware steering unit of a clustered microarchitecture that performs cluster hopping. Section 5 presents the performance results for the proposed techniques. Section 6 highlights the related work and Section 7 concludes the paper.

Proceedings of the IEEE International Conference on Computer Design (ICCD'04)
1063-6404/04 $ 20.00 IEEE

IEEE COMPUTER SOCIETY

2. Processor Architecture

2.1. Clustered Architecture

This Section briefly describes the baseline clustered microarchitecture.

The frontend reads IA32 instructions from the UL2, translates them into *micro-ops* and stores them in the trace cache, from where they are read, decoded, and steered to any of the backends, according to a steering policy. Each of the backends has its own register file, integer and floating point issue queues and a memory order buffer along with a data TLB and a first-level data cache. Because of limited space, the reader is referred to [7] where more details about the architecture are given.

2.2. Power and Temperature Model

The dynamic power model is very similar to those existing in the literature [4]. Leakage power has been modeled as the average dynamic power multiplied by a factor dependent on the temperature. More precisely, it is assumed that leakage power is going to be roughly 20% of dynamic power at ambient –inside box– temperature (45º [23]). This percentage is varied according to the current temperature of the functional block [7][25].

The temperature model is similar to the one by Skadron *et al.* [23]. It is based on the duality of the thermal and the electrical phenomena. The temperature is estimated using a RC model that represents the system, also known as dynamic compact model (dynamic because it includes thermal capacitors modeling the transient response of the system). At the microarchitectural level, it models heat conduction and the removal of heat in the heat sink.

3. Evaluation of the Clustered Microarchitecture

3.1. Experimental Framework

We have selected ten SPEC2000 applications for the evaluation process. Each execution trace (from the *test* input set) is divided in 10 equal-size slices (i.e., slices of different applications have different size) and the fourth of them is selected to be run (up to 250 millions of instructions).

Experiments have been conducted using an execution-driven simulator that runs IA32 binaries. Table 1 summarizes the main parameters of the simulated architecture. At the beginning of the simulation we assume that the processor has already been running for a long time dissipating its nominal average dynamic power and the leakage power at 63ºC. In this way, simulations are started with the processor already warmed. The simulator executes 1 million instructions and the following 9 million are fast-forwarded (the last million out of the nine is used to warm-up caches). At the end of the 10 million instructions interval, thermal sensors placed in all functional blocks are read and data is provided to the steering unit.. The interval is long enough (in the order of μs) so that the thermal sensors can stabilize and provide an accurate metric. At the same time, the interval is small enough so that our techniques can react to temperature changes.

Table 1. Processor configuration

Frontend	
12 cycles fetch, decode and steer (regardless of the destination cluster), fetch, dispatch and commit up to 8 *micro-ops* / cycle	
UL2	2 MB/8-way, 12 cycle hit, 500+ miss
Trace cache	32K*micro-ops*, 8-way, 4 cycle fetch-to-dispatch latency
Communi-cations	2 memory buses, 2 disambiguation buses, 4-cycle latency + 1-cycle arbiter. 2 bidirectional p2p link
Each Backend	
Queues	20-entry IQueue, 1 inst/cycle, 20-entry FPQueue 1 inst/cycle, 20-entry CopyQueue 1inst/cycle , 96-entry MemQueue 1inst/cycle, 10 cycle dispatch
Reg. File	160 int. registers and 160 FP registers, 6 read ports, 4 write ports
Data cache	16 KB/2-way, 1 cycle hit, 1 read port, 1 write port, writhe through with invalidation (no data transfer among)
Technology	
65nm, 10GHz, 1.1V, copper heat spreader, 3.1x3.1x0.23cm (Pentium® 4 [15]), copper heat sink of 7x8.3x4.11cm ([15]), 47 thermal sensors [22]	

The total area, computed using and enhanced version of Cacti ([21]) for cache-like structures, and scaling down rest of structures from current designs, is 101mm^2, which makes it feasible to be included in a bigger *CMP* configuration.

3.2. Baseline and Static Schemes

This Section shows the initial results for the baseline architecture and some minor modifications in order to gain insight into the behavior of temperature and leakage power.

Figure 1 shows the reduction of leakage power in the backend area (which represents around 20% of the total leakage of the processor), the reduction of various temperatures (in both cases higher is better) and the execution time slowdown (lower is better) for four different *non-thermal* steering schemes. Three

IEEE COMPUTER SOCIETY

different temperature metrics are shown (as the increase with respect to ambient temperature, 45ºC):

- *AVGTmp*: Average backend temperature (4 clusters)
- *MaxTmp*: Maximum backend temperature
- *AVGMaxTmp*: Average of the maximum temperature of the backend over time

Our baseline is a quad-cluster architecture with a workload- and communications-aware steering policy [6]. Some representative scenarios are shown: *Cluster 2* (or *0*) refers to a quad-cluster processor where instructions are just steered to cluster 2 (or 0), *Clusters 2 and 3* (or *0, 2 and 3*) refers to a configuration where just those clusters are eligible by the steering logic. Non utilized clusters do not dissipate any power.

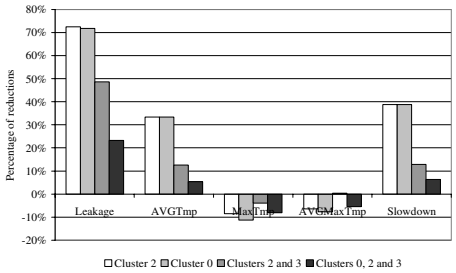

Figure 1. Baseline results

First, we can observe that *Cluster 2* or *Cluster 0* do not provide the same results in temperature and leakage due to the proximity of cluster 0 to some of the hottest areas of the frontend. Another interesting conclusion is that using a subset of clusters does not provide a reduction in maximum temperatures; on the contrary, they are even increased. The reason is that the work (and thus the dissipated power) is concentrated among a subset of the clusters. Moreover, the slowdown experienced by these coarse grain approaches is quite significant.

We have also run simulations for a monolithic superscalar processor with the same total resources as the whole clustered processor (80-entry IQueue, 80-entry FPQueue, 384-entry MOB, 4-way issue from each of the queues, etc). Comparing the performance against the monolithic processor, the non-thermal steering, clustered microarchitecture reduces the average temperature of the backend area by 20%, the peak temperature is reduced by 27% and leakage decreases by 27%, at the expense of a reduction in IPC of around 20% (delays and clock frequencies should be estimated to translate IPC into performance).

Finally, we can conclude that gating some of the clusters reduces leakage but there should be some way of rotation to distribute energy dissipation if we want to decrease temperature. Moreover, the steering unit must be aware the gating and try to reduce the slowdown combining thermal-aware and performance-aware policies.

4. Thermal-Aware Cluster Organizations and Steering

This Section presents our different proposals for a thermal-aware clustered microarchitecture. We investigate ways of reducing both leakage power (through temperature control) and maximum temperatures. These issues are attacked from two different angles:

- *Thermal-aware steering techniques*: we explore different alternatives to steer instructions to clusters according to their temperature.
- *Cluster-hopping*: we evaluate different configurations for a quad-cluster architecture where the set of active clusters dynamically changes.

Note that both approaches are complementary and can provide synergistic effects: a cluster-hopping architecture which disables some clusters will benefit from a temperature-aware steering scheme for dispatching instructions to the active clusters.

4.1. Temperature-Aware Steering Techniques

This Section presents our proposals for steering policies that attempt to reduce leakage and peak temperature. All policies presented in this paper rely on having temperature sensors placed in the functional blocks that provide the temperature at fixed intervals. Conceptually, the steering unit sorts all clusters according to a given policy and sends the instruction to the first cluster that has enough resources available (issue queue and register file entries)

The set of proposed steering techniques are the following:

- **Coldest cluster (*Cold*)**: given any two clusters, the one with lower temperature has priority to receive the instruction.
- **Coldest cluster with threshold (*T-Cold*)**: clusters are ordered following the previous policy. However, if the temperature difference between a cluster and the previous one in the sorted list (which is colder) is greater than a certain threshold, this and the following clusters (which are hotter) are not considered for dispatching the current instruction.
- **Temperature-, workload- and communication-aware (*T-Wload*)**: This policy trades-off communication, workload balance and temperature balance. For any pair of clusters (*Ci* and *Cj*) their temperature difference is computed (*difftemp = TempCi – TempCj*). If *difftemp* is greater than 0, it means that *Cj* is colder than *Ci*.

IEEE COMPUTER SOCIETY

Then, an imbalance threshold (*IT*) is computed as a function of *difftemp*. If the occupancy of *Ci* multiplied by *IT* is greater than the occupancy of *Cj*, then *Cj* is given more priority. If *difftemp* increases, *IT* also increases, so that *Cj* will have more probability of receiving the instruction. With this scheme, an instruction may be steered to a cluster that is not the least loaded one, depending on the temperature balance. The idea behind this heuristic is to calibrate workload imbalance according to temperature difference. If there is not a workload imbalance according to the above metric, the cluster holding most of the inputs is given priority.

- **Thermal threshold (T-Thermal)**: given two clusters, if the temperature difference is greater than a threshold, the coldest cluster is given priority to receive the instruction; otherwise, the one holding most of the inputs is given priority.

4.2. Cluster-Hopping

This architectural technique is based on using only a subset of the clusters for execution whereas the rest of the clusters remain inactive (V_{dd}-gated, so that they dissipate neither dynamic nor leakage power). In this way average temperature is reduced since the energy savings provided by disabling one or more clusters are greater than the increase on energy consumption experienced by the rest of active clusters, despite of the slight increase in their activity.

Before putting a cluster to sleep, the contents of its register file must be copied to the rest of clusters. A set of *copy micro-ops* is inserted in the cluster in order to copy the value of the logical registers whose latest mapping is not present in any other cluster. Each register value is sent to the nearest cluster. Since the decision of gating clusters is taken every 10M instructions, the performance impact of these copies is completely negligible. In addition, the contents of data cache and data TLB are lost (data caches are write-through, so next memory level always has an up-to-date copy).

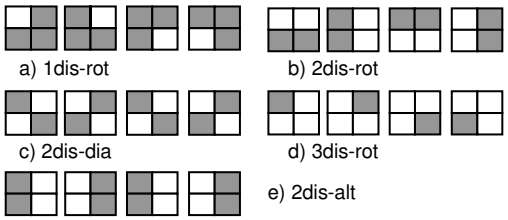

Figure 2. Cluster-hopping alternatives analyzed

Figure 2 graphically describes the different hopping alternatives evaluated in this work. White squares represent disabled clusters whereas gray squares represent active clusters. Each alternative consists of 4 intervals that are repeated for ever. For instance Figure 2a shows the basic hopping approach, where there is just one disabled cluster during each interval that rotates clockwise at the end of the interval.

Note that statically gating a certain amount of clusters or having an architecture with less clusters would result in a performance close to the one reported in Figure 1. Since activity would concentrate in less area, power density, leakage and temperature would increase. This problem is addressed by the hopping scheme.

5. Performance Evaluation

This Section evaluates a clustered architecture augmented with the thermal-aware steering techniques and the cluster-hopping schemes. The experimental methodology is the same as for the one reported in Section 3.1.

5.1. Analysis of the Thermal-Aware Steering

In this Section the thermal-aware steering techniques for a quad-cluster architecture without cluster-hopping are analyzed.

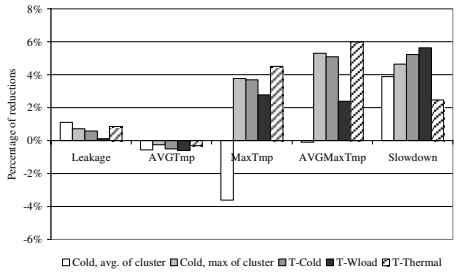

Figure 3. Performance of the thermal-aware steering techniques

Figure 3 shows some metrics for the thermal-aware steering policies presented in the previous Section. For the Cold policy, two implementations are presented, the first one sends the instructions to the cluster that has the coldest average temperature, and the second one steers the instruction to the cluster with the minimum peak temperature. It can be seen that using peak temperatures to make the decisions is more effective.

We can observe that *T-Thermal* is the best thermal-steering technique, since it provides the best thermal reductions and the lowest slowdown (just 2%). Second, the steering techniques proposed in this work are not very effective at reducing leakage power and average temperature (actually it is slightly increased). This is due to the fact that keeping all clusters active results in about the same dynamic activity and no leakage savings. However, they are quite effective to reduce both maximum temperature (4% average

IEEE COMPUTER SOCIETY

reduction and up to 7%). All analyzed applications follow a very similar trend.

5.2. Analysis of Cluster-Hopping

Figure 4 shows the performance of the cluster-hopping mechanisms, as described in Figure 2, when implemented with the baseline steering policy.

In general, cluster-hopping is effective at decreasing average temperature, since disabled clusters cool down through the heat spreader and absorb the heat of the active clusters. In addition the average of peak temperatures is also decreased for the same reason. On the other hand, cluster hopping is not very effective at reducing absolute peak temperature because of the concentration of activity in the enabled clusters.

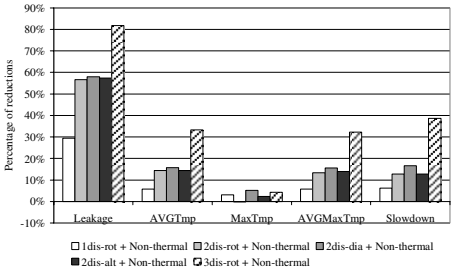

Figure 4. Performance of the cluster-hopping policies with the non-thermal steering

The difference between hopping two clusters clockwise or "diagonal" is not very significant. It is interesting to note that the performance of "diagonal" is a bit worse due to the increase in the latency of *copy* instructions generated for communicating register values. Note that data communication between adjacent clusters is faster than a communication between "diagonal" cluster, since data performs just one hop through the point-to-point network in the former case and two in the latter.

To conclude, the cluster-hopping policy that provides the best trade-off between performance and temperature/leakage reduction is *1dis-rot* (29% leakage reduction and 6% performance degradation).

5.3. Combining Thermal-Aware Steering Policies and Cluster-Hopping

Figure 5 shows the evaluation of an architecture that combines cluster hopping and thermal-aware steering for several different hopping schemes. For comparison, *T-Thermal* and *1-dis+Non-thermal* are also presented.

Combining cluster-hopping and *T-Thermal* outperforms both techniques when implemented individually. Disabling clusters increases power density in active clusters, which could increase peak temperature, but this increase is more than offset by the combination of cluster hopping and thermal-aware steering.

To conclude, the technique that provides the best trade-off between performance, leakage and temperature is *1disrot+T-Thermal,* which combines a thermal-aware steering policy with hopping. It obtains 8% reduction in average peak temperature, 6% decrease in average temperature, 5% reduction in peak temperature, 30% decrease in leakage and just a 5% slowdown.

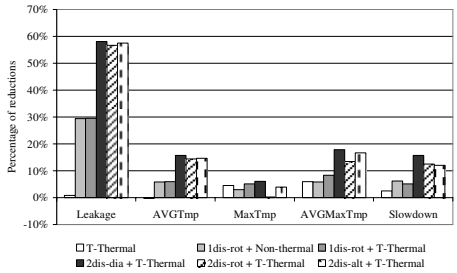

Figure 5. Performance of the best configurations combining thermal-aware steering policies and cluster hopping

6. Related Work

Controlling temperature through microarchitectural techniques is a fairly new area. Huang *et al* [14] propose a framework to maximize energy savings and to guarantee that temperature remains under a certain threshold. The framework combines a number of energy-management techniques. Brooks and Martonosi [5] propose a set of control techniques evaluated on top of different triggering mechanisms with the aim of reducing thermal emergencies. Skadron *et al.* [23] propose a thermal simulator based on the duality between heat transfer and the electrical phenomena. Several techniques are evaluated to control peak temperature and reduce thermal emergencies. Lim *et al* [18] propose a secondary ultra-low power pipeline that is used when a given temperature threshold is exceeded. Heo *et al.* [12] study the impact of activity migration among replicated units on power density.

Reducing leakage has received the attention of a plethora of studies, as for instance drowsy caches [10], leakage-biased bitlines [13], stacked gates [24], higher V_{th} [16], body bias [17], and others.

7. Conclusions

This paper has analyzed the thermal behavior of a clustered microarchitecture. First, we have proposed several thermal-aware instruction steering schemes. Second, we have analyzed different approaches in order to perform cluster-hopping in the processors' backend. This technique is based on disabling some particular clusters during a period of time and then

IEEE COMPUTER SOCIETY

rotating the disabled clusters to better distribute power dissipation. Finally, we have combined thermal-aware steering policies with different cluster-hopping schemes and shown that they have a synergistic effect.

Results show that thermal-aware steering is effective at reducing maximum temperature but it cannot control average temperature, since all units are kept active and dissipating power. On the other hand, cluster-hopping is an attractive option to control average temperature and leakage, but peak temperature is not reduced. Combining *T-Thermal* steering along with *1dis-rot* hopping provides the best results: average peak temperature is reduced by 8%, average temperature decreases by 6%, peak temperature is reduced by 5% and backend leakage is decreased by 30%, at the expense of a 5% slowdown.

An important feature of these techniques is that they are adaptive to workload features and environmental conditions. In addition, they are orthogonal to other adaptive power reduction techniques such as frequency and voltage scaling.

8. Acknowledgements

The authors would like to thank Kevin Skadron for his comments and suggestions on the thermal model.

9. References

[1] V. Agarwal, M.S. Hrishikesh, S.W. Keckler and D. Burger. "Clock Rate versus IPC: the End of the Road for Conventional Microarchitectures". In Proc. of the 27th Int. Symp. on Computer Architecture, 2000.

[2] R. Balasubramonian, S. Dwarkadas and D.H. Albonesi. "Dynamically Managing the Communication Parallelism Trade-off in Future Clustered Processors. ". Proceedigns of the Int. Symp. on Computer Architecture, 2003.

[3] S. Borkar. "Design Challenges of Technology Scaling". IEEE Micro, 19(4), pp. 23-29, 1999.

[4] D. Brooks, V. Tiwari and M. Martonosi. "Wattch: A Framework for Architectural-Level Power Analysis and Optimizations", in Proc. of the 27th Int. Symp. on Computer Architecture, pp. 83-94, 2000

[5] D. Brooks and M. Martonosi. "Dynamic Thermal Management for High-Performance Microprocessors". Proc. of the Int. Symp. on High-Performance Computing, 2001.

[6] R. Canal, J.M. Parcerisa and A. González. "Dynamic Cluster Assignment Mechanisms". In Proc. of the Int. Symp. on High Performance Computing. 2000.

[7] P. Chaparro, J. González and A. González. "Thermal-Effective Clustered Microarchitectures". In Proc. of the First Workshop on Temperatura Aware Computer Systems at ISCA 2004.

[8] V. De and S. Borkar. "Technology and Design Challenges for Low Power and High Performance". Proc. of the Int. Symp. on Low Power Electronics Design pp. 163-168, 2000.

[9] K. Farkas, P. Chow, N. Jouppi and Z. Vranesic. "The Multicluster Architecture: Reducing Cycle Time through Partitioning". Proc. of the Int. Symp. on Microarchitecture, 2000.

[10] K. Flautner, N.S. Kim, S. Martin, D. Blaauw and T. Mudge. "Drowsy Caches: Simple Techniques for Reducing Leakage Power". Proc. of the Int. Symp. on Computer Architecture, 2002.

[11] S. Gunther, F. Binns, D. M. Carmean and J.C. Hall. "Managing the Impact of Increasing Microprocessor Power Consumption". Intel Technology Journal, Q1, 2001.

[12] S. Heo, K. Barr, K. Asanović "Reducing power density through activity migration" Low Power Electronics and Design, 2003. ISLPED '03. Proc. of the 2003 Int. Symp. on, 25-27 Aug. 2003.

[13] S. Heo, K. Barr, M. Hampton and K. Asanovic. "Dynamic Fine-Grain Leakage Reduction Using Leakage-Biased Bitlines". Proc. of the Int. Symp. on Computer, 2002.

[14] M. Huang, J. Renau, S-M. Yoo and J. Torrellas. "A Framework for Dynamic Energy Efficiency and Temperature Management". Proc. of the Int. Symp. on Microarchitecture, pp. 202-213, 2000

[15] Intel Corporation "Intel® Pentium ® 4 Processor in the 423-pin Package Thermal Solution Functional Specification" http://www.intel.com/design/pentium4/guides/249204.htm.

[16] J.T. Kao and A. P. Chandrakasan. "Dual-Threshold Voltage Techniques for Low-Power Digital Circuits". IEEE Journal of Solid State Circuits, 37(5), pp. 1009-1018, 2000.

[17] A. Keshavarzi, S. Ma, S. Naredra, B. Bloechel, K. Mistry, T. Ghani, S. Borkar and V. De. "Effectiveness of Reverse Body Bias for Leakage Control in Scaled Dual Vt CMOS ICs." Proc. of the Int. Symp. on Low Power Electronics Design", pp. 207-212, 2001.

[18] C. H. Lim, W. R. Daasch, G. Cai, "A Thermal-Aware Superscalar Microprocessor" Proc. Int. Symp. on Quality Electronic Design, 18-21, 2002

[19] R. Majan "Thermal management of CPUs: A perspective on trends, needs and opportunities", Oct. 2002. Keynote presentation, THERMINIC-8.

[20] J.-M. Parcerisa, J. Sahuquillo, A. González, J. Duato "Efficient Interconnects for Clustered Microarchitectures" Proc. of the Int. Conf. on Parallel Architectures and Compilation Techniques (PACT 2002).

[21] P. Shivakumar, N. P. Jouppi "CACTI 3.0: An Integrated Cache Timing, Power and Area Model" WRL Research Report 2001/2.

[22] B. Sinharoy "POWER5 Architecture and Systems", Feb. 2004. Keynote presentation, Int. Symp. on High Performance Computer Architecture

[23] K. Skadron, M. R. Stan, W. Huang, S. Velusamy, K. Sankaranarayanan, and D. Tarjan. "Temperature-Aware Microarchitecture". In Proc. of the 30th Annual Int. Symp. on Computer Architecture, Apr. 2003.

[24] Y. Ye, S. Borkar and V. De. "A Technique for Standby Leakage Reduction in High-Performance Circuits". Proc. of the Symp. on VLSI Circuits, pp. 40-41, 1998.

[25] Y. Zhang, D. Parikh, K. Sankaranarayanan, K. Skadron and M. Stan. "Hotleakage: A Temperature-Aware Model of Subthreshold and Gate Leakage for Architects". T. R. CS-2003-05, Univ. of Virginia Department of Computer Science, Mar. 2003.

IEEE COMPUTER SOCIETY