History & Variation Trained Cache (HVT-Cache): A Process Variation Aware and Fine Grain Voltage Scalable Cache with Active Access History Monitoring

Avesta Sasan[1], Houman Homayoun[2], Kiarash Amiri[1], Ahmed Eltawil[1], Fadi Kudahi[1]
[1]Dept. of Electrical and Computer Engineering, University of California Irvine
[2]Dept. of Computer Science and Engineering, University of California, San Diego
mmakhzan@uci.edu, hhomayou@uci.edu, kamiri@uci.edu, aeltawil@uci.edu, kurdahi@uci.edu

Abstract

Process variability and energy consumption are the two most formidable challenges facing the semiconductor industry nowadays. To combat these challenges, we present in this paper the "History and Variation Trained-Cache" (HVT-Cache) architecture. HVT-Cache enables fine grain voltage scaling within a memory bank by taking into account both memory access pattern and process variability. The supply voltage is changed with alterations in the memory access pattern to maximize power saving, while assuring safe operation (read and write) by guarding against process variability. In a case study, SimpleScalar simulation of the proposed 32KB cache architecture reports over 40% reduction in power consumption over standard SPEC2000 integer benchmarks while incurring an area overhead below 4% and an execution time penalty smaller than 1%.

Keywords

Low power memory design, process variation, low power, voltage scaling, reconfigurable cache, process variation aware cache

1. Introduction

Probably the most limiting concern for advancement of our field is the issue of increased power density in scaled technologies. On one hand, the higher frequency of operation mandates larger dynamic power consumption. On the other hand, the increased leakage in scaled technologies, combined with market, application and performance driven demand of having larger memory structures on chip has increased the contribution of static power towards a chip's total energy consumption. In fact, static power is on the verge of dominating the dynamic power consumption [1][2][3][10][17][20].

A very effective knob to manage and reduce power consumption is voltage scaling, as both dynamic and static components of power consumption are super-linearly reduced by a linear reduction in the supply voltage. However application of voltage scaling to memory structures not only reduces the operational speed, but also raises reliability issues, which are exacerbated by process variation. Increased process variability in scaled technologies had reduced the reliability and predictability of electrical and logical characteristics of manufactured devices [3][4][5]. Due to introduced variability, the write and access time of the memory can be modeled as a Gaussian distributions. Not only does voltage scaling shift the mean of access/write time, but it also changes the standard deviation of those distributions [6].

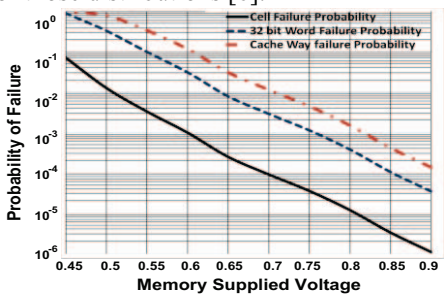

Figure 1: Probability of cell, way and cache failure in 32nm technology. for a 32KB 8 way associative cache organizations.

Figure 1 depicts the results of a Monte Carlo simulation for a 6T SRAM cell under process variation in 32nm technology (with standard deviation of 34mV for the threshold voltage [6]). The Figure illustrates the exponential growth in the probability of cell failure with a reduction in supplied voltage. In obtaining this curve, cycle time is kept constant (to that of used in higher voltage). Depending on the choice of cycle time, different probability of failure curves can be obtained.

In this paper, we propose a novel cache architecture that fine-tunes itself to get the maximum power saving trough adaptive and fine grain voltage scaling while accounting for process variability. This structure takes advantage of a simple and low cost distributed supply voltage management that allows a majority of the memory cells to safely operate at a reduced supply voltage. In this design, a cell that is severely affected by process variation does not dictate a larger voltage to the entire cache (i.e. the minimum "safe" supply voltage, V_{CC}^{Min} , is not mandated by the weakest cell). Instead, the higher voltage requirement is only mandated in the cache way(s) that contains the weak cell(s). The proposed cache architecture explores the access history of each set in the cache to supply some weak cells from a lower voltage for as long as they are not involved in a memory operation.

2. Prior Work

In the recent years there has been a flurry of research activity to manage process variation and/or power consumption of memories in general and caches in particular [7][8][13][15][16]. In [7][8], cache lines that are not recently accessed are power gated. When a gated line is accessed, it is

charged back to nominal voltage, which requires charging all the internal capacitances of the memory cells in that cache line. Furthermore, the next level cache should be accessed to retrieve the information. [15] proposes MC^2 which maintains multiple copies of each data item, exploiting the fact that many embedded applications have unused cache space resulting from small working set sizes. On every cache access, MC^2 detects and corrects errors using these multiple copies. Thus MC^2 – while particularly useful for embedded applications with small working sets – may result in high area and performance overhead for other applications, particularly in the presence of high fault rates. In [16] RDC-cache is proposed which replicates a faulty word by another clean word in the last way of next cache bank. In [13] FFT-Cache is proposed which uses a portion of faulty cache blocks as redundancy using block-level or set-level replication within or between sets to tolerate other faulty caches sets and blocks.

In [9] an Inquisitive Defect Cache (IDC) is presented, which is used as a cache that works in parallel with L1 cache and provides a defect-free view of the cache for the processor. This technique reduces the voltage on the entire cache and maps the faulty cache ways which are recently accessed to the IDC that operates at nominal voltage. Although the proposed architecture achieves considerable saving in power consumption, the associated area overhead is not negligible. A recent paper from Intel's microprocessor technology lab [10] suggests trading off the cache capacity and associativity for masking process variation defects. The proposed approaches allow scaling the voltage while the cache size is reduced by 75% or 50% depending on the fault tolerance mechanism used. This technique is used whenever the processor workload is low. As will be described in the paper, the proposed HVT-Cache can be used both at nominal frequencies as well as at reduced workloads, while maintaining the maximum cache size at reduced power consumption.

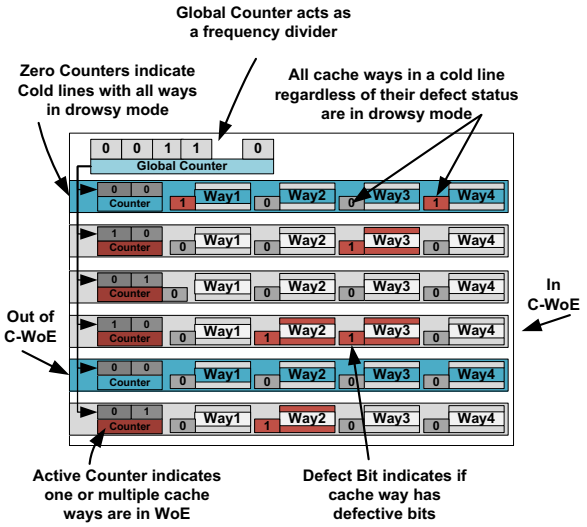

Figure 2: Top level view of the HVT-Cache; Sets with non-zero counters are in the WoE. Any cache way that contains a weak cell (defect bit =1) and is located in one of the sets within WoE is sourced with V_{dd}^{High}.

3. Proposed Architecture: HVT-Cache

3.1 Concept:

The HVT-Cache enables fine grain voltage control on way granularity. HVT-Cache chooses one of the two voltage levels to supply the cache way using a simple voltage selector that is implemented at each cache way. The voltage selector dynamically chooses between the two states as the processor executes new segments of the running program, and shifts (and/or resizes) its Window of Execution in the cache (C-WoE). C-WoE is defined as the cache ways that are accessed within the previous N cache accesses. In HVT-Cache, a low supply voltage (V_{dd}^{Low}) is selected when either: a) the cache way in not in the C-WoE, or b) the cache way is in C-WoE but at the given memory cycle time all memory cells in that cache way could be read and written at the lower supply voltage. If none of these conditions is met, the cache way is supplied with V_{dd}^{High}. The HVT-Cache explores power saving opportunities by applying predictive fine grain voltage scaling based on access history. The access history is logged for each cache set using a low overhead mechanism which will be discussed shortly. The decision to use which supply voltage is made based on a defect map that is generated using memory Build-In Self-Test (BIST).

Figure 2 outlines the top view of HVT-Cache organization. Each set spans 4 ways and has a dedicated Set Access Manager (SAM). The SAM has an internal N (usually $1 \leq N \leq 3$) bit counter. Upon access to any cache way in that set, the set is identified as being in the C-WoE by setting the SAM counter to a nonzero value. Within the set, each cache way has its own simple and dedicated Way Voltage Selector (WVS), which is linked to a defect bit that indicates whether or not that way contains one or more defective bitcells. If the SAM counter reaches zero, all WVSs that are associated with that SAM force the state of their cache way to V_{DD}^{Low}. Otherwise, the WVSs , supply the cache way from either V_{DD}^{High} or V_{DD}^{Low} depending on whether or not there are defects in that way, as indicated by the defect map. The SAM counter counts down when a Count Down Signal (CDS) is pulsed by the global counter. The global counter acts as a cache access frequency divider and is shared among all the sets. It is a cyclic counter that counts down and generates the CDS-signal upon reaching 0 while being reset to its high value. The number of bits in local set counters and global counter affects both performance and power consumption as will be discussed in Section III.C

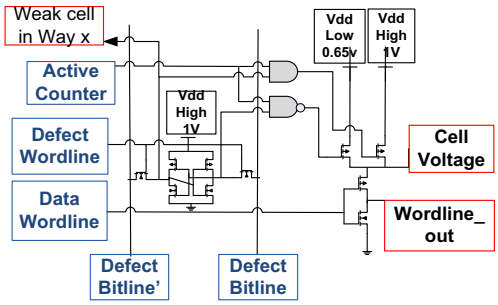

Figure 3: cache Way Voltage Selector (WVS)

3.2 Implementation:

Way Voltage Selector (WVS) is shown in Figure 3. It contains an internal memory bit referred to as Fault Tolerant Bit (FT-Bit). FT-Bit is set if the cache way contains weak memory cell(s) that are severely affected by process variation such that they malfunction at V_{dd}^{Low}. The implemented FT-Bit is made more tolerant to process variation by upsizing the basic 6T cell, or by using a Schmitt Trigger Cell [11]. It is updated after running a BIST at low voltage and is written by using the same mechanism as other SRAM cells using dedicated Defect Bitlines as illustrated in Figure 3. The "Defect Wordline" input is derived by the SAM when the system requires updating the defect map.

In this paper we introduce two different versions of the HVT-Cache: one with larger area but lower power consumption referred to as Blocking-HVT-Cache, and the other with smaller area and slightly larger power consumption named Inquisitive-HVT-Cache. The implementation of the SAM separates the two implementations. Before introducing these entities we need the definition of a soft miss. A soft miss is a cache access that cannot be granted due to presence of weak cells that are supplied from a lower voltage level. This happens when a set outside of C-WoE is accessed. The two variations of HVT-Cache differ by the policy that generates a soft miss. The Blocking-HVT-Cache declares a soft miss any time there is an access to a set outside the C-WoE that contains at least one cache way with one or more weak cells. In this case the SAM counter is set causing the weak cache way to be sourced from the higher supply voltage, and the cache access is repeated. The implementation of SAM for Blocking-HVT-Cache is illustrated in Figure 4

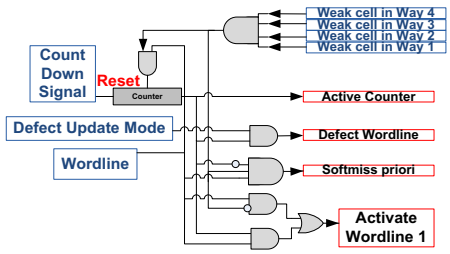

Figure 4: SAM in Blocking-HVT-Cache

In Inquisitive-HVT-Cache, the SAM performs a bit more when such row is accessed, thereby allowing better performance and possibly lower energy consumption at the expense of larger area overhead. SAM for Inquisitive-HVT-Cache is illustrated in Figure 5. The SAM blocks the access to weak cache way(s) and generates a signal called "soft miss apriori" while allowing access to healthy cache ways and the TAG to continue. If there is a cache hit, the soft-miss apriori signal of that cache way is checked. If the signal is raised a soft-miss is generated otherwise the data should be ready to be read.

In both implementations the SAM contains an internal counter which counts down every time CDS signal is pulsed. The "Defect Update Mode" signal is raised whenever the system desires to change the HVT-Cache defect map. This signal is input to all SAMs. Once this signal is raised, a rise in the wordline will activate the "defect-wordline" output, allowing the update of the defect bit implemented at each WVS. The Active Counter output is raised if the counter is non zero. This output is used by SAM and WVSs to determine if the cache way should be supplied from V_{dd}^{Low} or V_{dd}^{High}.

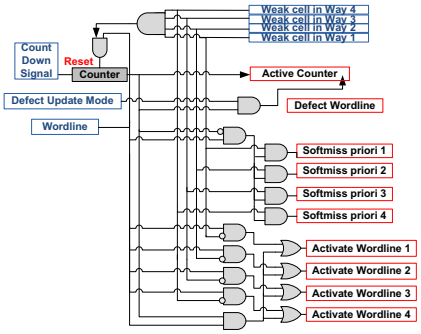

Figure 5: SAM in Inquisitive-HVT-Cache

CDS in HVT-Cache is generated using a global access counter (or frequency divider). The global counter logically extends the LSBs of all SAM local counters. This mechanism allows the HVT-Cache to estimate the window of execution with a much smaller overhead compared to implementing a full counter for each set. Since the global counter is a cyclic counter that sends the CDS signal to the local set counters every time it reaches the 0 state, its length (i.e. the global counter) determines the frequency of updates to those set counters.

3.3 Accuracy of Prediction of Cache Window of Execution

Upon access to a set, its SAM's counter is set. At this time the global counter could have any value. Therefore the

accuracy of the extended logical counter (with SAM counter at MSB and global counter at LSB) is only controlled by the initial value of the SAM counter while the global counter introduces uniform randomness in the LSBs. Because it is shared, the global counter introduces a negligible area overhead while the SAM counters' area overhead (repeated for every set) could be significant. Choosing the right "split point" between the two slices is therefore a tradeoff between accurately estimating the C-WoE and area overhead. In HVT-Cache this tradeoff is explored by carefully sizing the local and global counters. Having 'm' local bits, the inaccuracy in determining the C-WoE is $\frac{1}{2^m}$ meaning the starting point of the extended counter could range from $[2^{m+n}-1\,,2^{m+n}-2^n-1]$. For example, in an architecture with a 2-bit local and 7-bit global counter the logical counter upon access could be set to a max value in the range of $[2^{2+7}-1\,,2^{2+7}-2^7-1]$ or [511, 383].

3.4 A Model to Measure Energy Consumption

In this section we explain our model for calculating the energy consumption of the HVT-Cache for different benchmarks. The dynamic and static energy consumption of the HVT-Cache and conventional cache are obtained from SPICE simulation of the post layout netlist of these caches and is used in this model. In addition information on type, number and nature of accesses to the cache for different benchmarks is obtained using SimpleScalar [12] simulation after we modified SimpleScalar to model the HVT-Cache. For simplicity in this model we assumed that the TAGs are supplied from V_{dd}^{High}.

The energy improvement metric is thus calculated as follows:

$$\text{Percentage Improvement} = \left[1-\frac{E_{total}^{VT}}{E_{Total}^{Conv}}\right]*100 \quad (1)$$

The Energy consumption of HVT-Cache could be divided into dynamic and Static energy consumption:

$$E_{total}^{HVT} = E_{dynamic}^{HVT} + E_{static}^{HVT} \quad (2)$$

The dynamic energy consumption could be further broken down to that of Peripheral, Tags and Ways:

$$E_{dynamic}^{HVT} = E_{Peripheral}^{HVT-dyn} + E_{Tags}^{HVT-dyn} + E_{Ways}^{HVT-dyn} \quad (3)$$

The dynamic energy consumption is divided to the energy for reading and writing the memory cells.

$$E_{Tags}^{HVT-dyn} = E_{Tags-Read}^{HVT-dyn}.\left[N_{read}^{High} + N_{Read}^{Soft-Miss}\right] + E_{Tags-Write}^{HVT-dyn}.\left[N_{Write}^{High} + N_{write}^{Soft-Miss}\right] \quad (4)$$

$$E_{Ways}^{HVT-dyn} = E_{Ways-Read}^{HVT-dyn}.\left[N_{read}^{High} + N_{read}^{Low}*\left(\frac{Vdd_{low}}{Vdd_{high}}\right)^2\right] + E_{Ways-Write}^{HVT-dyn}.\left[N_{Write}^{High} + N_{Write}^{Low} *\left(\frac{Vdd_{low}}{Vdd_{high}}\right)^2\right] + E_{Transition}^{HVT-dyn} \quad (5)$$

The $E_{Transition}^{VT-dyn}$ in Equation (5) is the energy consumed when changing a cache way from low to high voltage accounting for energy spent in charging the internal capacitances and is calculated based on:

$$E_{Transition}^{HVT-dyn} = N_{High}^{Low}.E_{Transistion-to-high-voltage}^{HVT-dyn} \quad (6)$$

In which N_{High}^{Low} is the number of low-to-high transitions. The peripheral energy consumption is also divided by peripheral energy consumption for reads and writes:

$$E_{Peripheral}^{HVT-dyn} = E_{Peripheral-read}^{HVT-dyn}*\left(N_{read}^{High}+N_{read}^{Low}+\theta.N_{read}^{Soft-Miss}\right) + E_{Peripheral-write}^{HVT-dyn}*\left(N_{write}^{High}+N_{write}^{Low}+\gamma.N_{write}^{Soft-Miss}\right) \quad (7)$$

In which θ and γ are the correction factors used to account for change in energy consumption during a soft miss in a read or write operation accordingly. The static power consumption of the HVT-Cache on the other hand is broken into the static power consumption of the Cache Ways, Tags and the Peripheral:

$$E_{Static}^{HVT} = E_{Peripheral}^{HVT-static} + E_{Ways}^{HVT-static} + E_{Tags}^{HVT-static} = N_{execution}^{Cycle}.\frac{1}{f}\left[P_{Peripheral}^{HVT-static} + P_{Ways}^{HVT-static} + P_{Tags}^{HVT-static}\right] \quad (8)$$

To simplify the analysis the temperature variation effect on static power consumption is neglected (assuming operation in constant temperature) and the static power consumption of the Cache ways is calculated by measuring the length of time which a cache way has been in low voltage or high voltage states.

$$P_{Ways}^{HVT-static} = P_{Ways-VDD_{High}}^{HVT-static}*\left[AVG_{High} + (N_{wys}-AVG_{High})*e^{b(Vdd_{low}-Vdd_{high})}\right] \quad (9)$$

The conventional cache power consumption is also needed in equation (1) and is obtained based on equation (10) by breaking the power consumption into dynamic and static power consumption.

$$E_{Total}^{Conv} = E_{dynamic}^{Conv} + E_{static}^{Conv} \quad (10)$$

The Static power consumption is obtained from:

$$E_{static}^{Conv} = N_{execution}^{Cycle} . \frac{1}{f} . P_{static}^{Conv} \quad (11)$$

And the dynamic power consumption is obtained from

$$E_{dynamic}^{Conv} = N_{Read} . E_{Read}^{Conv-dyn} + N_{Write} . E_{Write}^{Conv-dyn} \quad (12)$$

The V_{dd}^{Low} is chosen such that most of the cache ways within the active lines (C-WoE) are still readable. Choosing a V_{dd}^{Low} that is too low results in: a) an increase in the number of ways within the C-WoE that are supplied with higher voltage due to increase in the cell failure probability, b) an increase in energy required for transition of low to high voltage, and c) a rise in the execution time due to an increase in the soft-misses associated with a slower transition time and a higher failure rate. On the other hand, if V_{dd}^{Low} is chosen inappropriately large, the cache consumes higher dynamic power. In this case, the number of cache way that is supplied from V_{dd}^{High} is reduced, but we have to supply all the other healthy cache ways from a higher V_{dd}^{Low}.

3.5 Defect Map, BIST and Temperature Variation

For each functional setting (voltage, temperature and frequency) the defect map could be different. A simple solution is using the worst case defect map for safe operation by running the BIST at low voltage and highest temperature. However, such a pessimistic approach results in a waste of power during operation in nominal operational setting.

Many modern processors today are equipped with Digital Temperature Sensors (DTS) [13][14]. DTS allows the usage of operational region dedicated defect map rather than a worst case defect map. The generation, update and switching between defect maps with consideration for temperature variation is done as follows: a) After manufacturing and during functional testing, the cache is stress-tested for the highest possible temperature. Manufacturing defects and process variation defects that still malfunction at the highest voltage and highest temperature are redirected to available redundancy. b) The stress test (at high temperature) is repeated for V_{DD}^{Low} and the worst case defect map for the HVT-Cache is generated. c) At the first boot of the system, the HVT-Cache is loaded with the worst case defect map populated at manufacturing (step 2). d) The range of possible temperature variation is divided into different regions (each region covering a range of temperatures) and BIST is used to generate a defect map for each region; when temperature passes a region boundary for which a defect map does not yet exist, the BIST is executed and a new defect map is generated.

The populated defects maps are stored in non-volatile memory (e.g. Flash or H.D.D). Each time that the temperature enters a new boundary, the defect map of that region is loaded into HVT-Cache.

4. Area Overhead

Compared to a traditional cache, the HVT-Cache area overhead is introduced by: a) WVSs, which is repeated for each cache-way, b) SAMs, which is shared among cache ways in each set, c) enhanced comparators, and d) the global counter. In addition, using multiple supply voltages imposes extra routing overhead and complexity. To reduce the area overhead of the WVSs, the N-wells of pull-up transistors are shared and Well is pinned to the highest voltage. Sharing of N-wells reduces the drive power of PMOS transistors in the lower voltages. Our simulation revealed that the effect on read timing and failure probability is negligible. However due to higher dependency on PMOS transistor drive power, the write operation is negatively affected. In order to improve the write time, the write circuit drive strength is increased by widening its size (~ 10% increase).

Compared to a conventional cache realized using the same layout rules, The Blocking and Inquisitive HVT-Cache incurred 3.96% and 5.06% area overhead accordingly when realized in a 32KB, 4 way associative L1 data cache arranged in 2 banks.

In our Blocking-HVT-Cache layout roughly 57 percent of the area overhead is contributed from WVSs, around 21 percent comes from SAMs, and the rest is from routing (~17%), global counter (~5%) and enhanced comparators (negligible). In the Inquisitive-HVT-Cache the SAM area overhead is about 45% of the introduced area overhead.

Table1: SimpleScalar configuration

Parameter	Value
ROB size	256
Register File Size	256 FP, 256 INT
Fetch/schedule/retire/width	6/5/5
Scheduling Window Size	32FP, 32 Int, 32 Mem
Memory Disambiguation	Perfect
Load/Store Buffer Size	32/32
Branch Predictor	16KB
Cache Line Size	64 Byte
L1 Data Cache Size	32 KB, 4Way, 1 Cycles
L1 Instruction Cache Size	32 KB, 4Way, 1 Cycles
Execution Length	1B Fast Forward, 1B execution
L2 Unified Cache	2MB, 8Way, 6 Cycles

5. Results

5.1 Case Study: Finding the Optimal V_{dd}^{Low} and Width of Local and Global Counters

We use the model described in Section III.E to find the optimal sizes of local and global counters for a 32KB, 4 way associative L1 data cache arranged in 2 banks using the simpler SAM manager in a Blocking-HVT-Cache. Each cache way contains 4 words. The mapping of voltage to failure probability is provided in Figure 1. We simulated the architecture for different voltages and for different combination of local and global counters. The local counter is varied from 1 to 3 bits and the global counter from 3 to 9 bits

and finally the voltage is varied from a nominal 0.9 V to 0.6V. In this simulation based on mapping of voltage to failure probability in Figure 1, for each voltage defective cache ways are uniformly and randomly distributed in the cache. The SimpleScalar configuration is documented in Table 1. After fast forwarding 1 Billion instructions, the integer benchmarks are executed for 1 Billion instructions to extract the parameters needed for equations 1-12. The simulation is repeated 300 times for each benchmark each time using a different seed for distribution of the faulty cache ways (thus generating different defect maps).

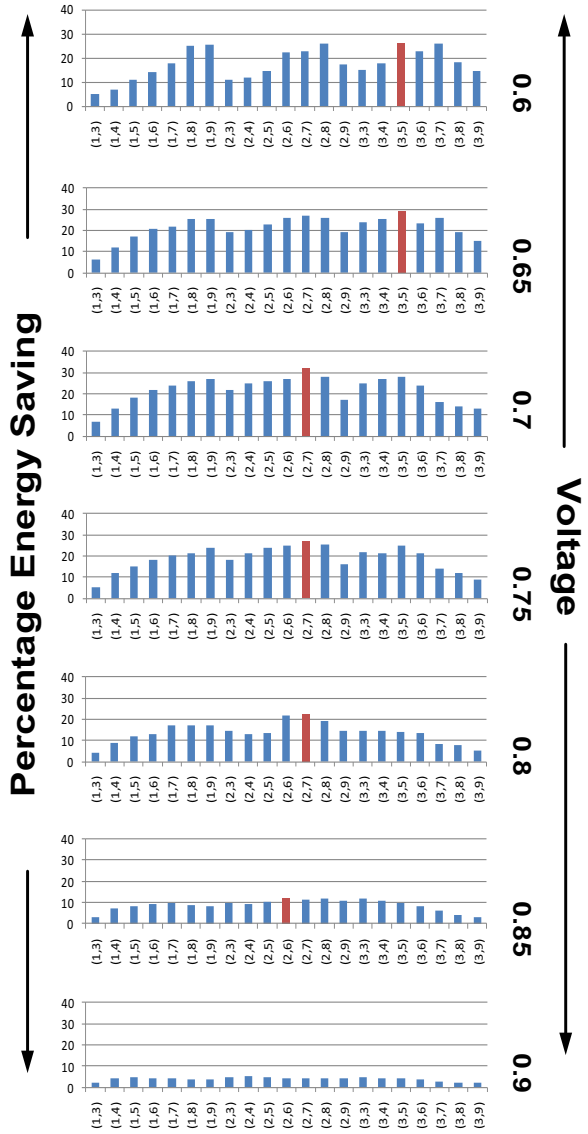

Figure 6: Percentage improvement in total energy consumption averaged over all integer benchmarks. Each bar represent the percentage energy saving of the combination of local and gloabal counter setting (local, global) at that voltage.

At each voltage the simulation is repeated for different choices of global and local counters. Based on the extracted parameters the improvement in total energy (based on equation 1) is obtained. Then the improvement index for each pair of local and global counter setting is averaged over all benchmarks and all runs. Figure 6 illustrates the obtained average energy improvement. This Figure suggests that for the given cache organization, at 0.7V, when a 7 bit global counter & 2 bit local counters is used, the power saving is maximized. Same results are obtained when the Inquisitive-HVT-Cache is simulated. The transition penalty of changing voltage from low to high is assumed to be one cycle.

5.2 Case Study: Energy Saving comparison between Inquisitive and Blocking HVT-Cache

In the following case study Inquisitive and Blocking HVT-Cache are simulated and compared. A setting of 2-bit local and 7-bit global counter is used. The energy model previously developed was used to calculate the energy consumption. Voltage scaling could be achieved using a wide range of policies that map each voltage to a frequency. In this paper, we purposely selected an aggressive model of voltage scaling in which--in order to keep the peak performance, the frequency is kept constant while voltage is scaled. This model is referred to as Fixed Frequency Voltage Scaling (FFVS). Adopting FFVS results in an exponential increase in the number of failures as voltage is scaled. Conventionally voltage scaling is applied when the processor workload is not high, and performance degradation is not an issue. Although HVT-Cache could also be used this way, but by adopting FFVS policy we intend to show that HVT-Cache could be used when near-peak performance is expected.

Figure 7 compares the energy savings between Inquisitive and Blocking HVT-Data-Cache for selected SPEC2000 benchmarks. Chosen benchmarks are selected carefully to represent different behavior of data access by SPEC2000 benchmarks. As Figure 7 suggests the Inquisitive-HVT-Cache in all cases results in better energy savings compared to the Blocking-HVT-Cache. In addition, and as suggested, some benchmarks better utilize the HVT-Cache compared to others. This is the result of varying grade of locality in these benchmarks. Benchmarks that have smaller and longer executing loops result in better energy savings and their behavior is better predicted by the access prediction mechanism of the HVT-Cache. In real time, the average number of ways that are not in a low voltage state varies depending on benchmark properties at that execution window. Typically the C-WoE is the largest during phase changes.

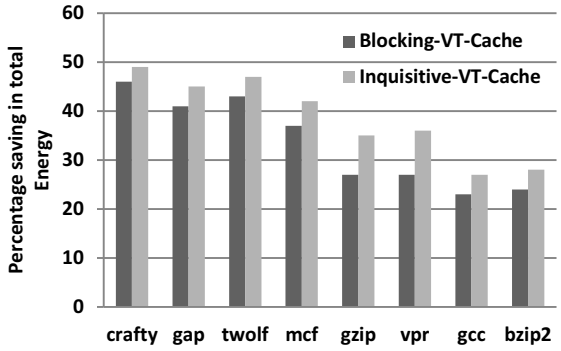

Figure 7: Comparing the improvement in total energy consumption between Inquisitive and Blocking HVT-Cache

Figure 8 compares the increase in the execution time of the Blocking and Inquisitive-HVT-Cache. The Inquisitive-HVT-Cache always result in lower execution time. This is the result of reduction in the number of soft-misses by using a more complicated SAMs. The percentage increase in execution time is related to many factors such as: 1) Penalty for a soft-miss due to transition between V_{dd}^{High} and V_{dd}^{Low}: The larger the associated penalties, the larger the execution time, 2) Locality of access to data and instruction: A higher locality reduces the chances of soft-miss thereby decreasing the number of transitions, and 3) Miss rate: Since the tagss supply voltage is not scaled in this architecture , and upon a miss on a drowsy line, (as long as the cache line is not accessed during access to L2 cache, which is assumed to be non-blocking), the line at the low voltage has the entire duration of L2 access to charge up to the writable voltage level without affecting the execution time. In addition, since the penalty of soft-miss compared to cache miss is small, having a lot of cache misses reduces the percentage contribution of soft-misses to the execution time.

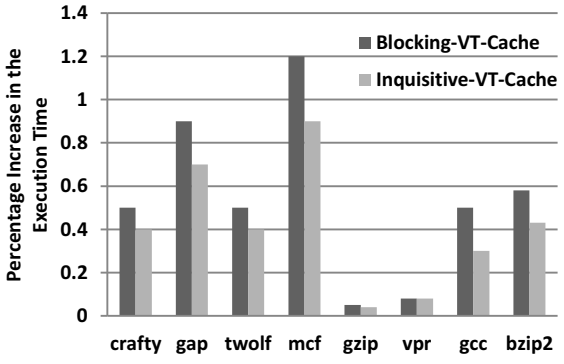

Figure 8: Comparing the execution time between Inquisitive and Blocking HVT-Cache

6. Conclusion

In this paper, we presented the History and Variation Trained Cache (HVT-Cache), which is a novel low power cache for high performance processors, while addressing the reliability issues raised by process variability. We explored the design space of the HVT-Cache architecture and its components. We demonstrated how the HVT-Cache setting (number of local bits, global bits and V_{dd}^{Low}) is chosen to maximize the improvement in total energy savings. Our simulation results indicate a significant improvement in total energy consumption across simulated benchmarks. While taking into account "weak cells", the HVT-Cache reduces dynamic the power consumption of accessing most of the cache ways within C-WoE. It also reduces the static power consumption of cache all ways supplied from the lower voltage. In future work, we will address the problem of enforcing triple voltage supply policy to tag section of the cache as well as dynamic reconfiguration policies and design issues to further improve energy consumption for adapting with changes in the phase of each benchmark execution.

7. References:

[1] Tsai, Y.-F.; Duarte, D.; Vijaykrishnan, N.; Irwin, M.J., "Implications of technology scaling on leakage reduction techniques," Design Automation Conference, 2003. Proceedings , vol., no., pp. 187-190, 2-6 June 2003

[2] Anis, M., "Subthreshold leakage current: challenges and solutions," Microelectronics, 2003. ICM 2003. Proceedings of the 15th International Conference on , vol., no., pp. 77-80, 9-11 Dec. 2003

[3] H. Homayoun, S. Pasricha, M.A. Makhzan, A. Veidenbaum: Dynamic register file resizing and frequency scaling to improve embedded processor performance and energy-delay efficiency, Proc. 45th ACM/IEEE Design Automation Conference DAC 2008, 2008.

[4] Bhavnagarwala, et. al. " The impact of intrinsic device fluctuation on CMOS SRAM cell stability," IEEE J. Solid-State Circuits vol.36, no.4 pp 658-665 Apr 2001\

[5] Sasan A. et. al., Limits on Voltage Scaling for Caches Utilizing Fault Tolerant Techniques", ICCD 2006.

[6] Sasan A. et. al., "Process Variation Aware SRAM/Cache for aggressive Voltage-Frequency Scaling" DATE 2009

[7] S. Kaxiras, Z. Hu, and M. Martonosi. Cache decay: Exploiting generational behavior to reduce cache leakage power. Proc. of Int. Symp. Computer Architecture, 2001, pp. 240- 251.

[8] H. Zhou, et. al. Adaptive mode-control: A static-power-efficient cache design. Proc. of Int. Conf. on Parallel Architectures and Compilation Techniques, 2001, pp. 61-70.

[9] A. Sasan, H. Homayoun, A.M. Eltawil, and F.J. Kurdahi. Inquisitive Defect Cache: A Means of Combating

Manufacturing Induced Process Variation. IEEE Transactions on VLSI Systems, 18(12):1-13, Aug. 2010.

[10] H. Homayoun, Mohammad Makhzan, Alex Veidenbaum, "Multiple sleep mode leakage control for cache peripheral circuits in embedded processors", in Proc. CASES 2008.

[11] J. P. Kulkarni, et. al., "A 160 mV Robust Schmitt Trigger Based Subthreshold SRAM,," IEEE Journal off Solid-state Circuits, Vol.. 42, no.. 10, pp. 2303-2313, October, 2007.

[12] http://www.simplescalar.com/

[13] A. BanaiyanMofrad, H. Homayoun, N. Dutt: FFT-cache: a flexible fault-tolerant cache architecture for ultra low voltage operation. Proceedings of the 14th International Conference on Compilers, Architecture, and Synthesis for Embedded Systems, CASES 2011 ,pp.95-104.

[14] http://www.intel.com/technology/itj/2006/volume10issue02/art03_Power_and_Thermal_Management/p03_power_management.htm.

[15] A. Chakraborty, H. Homayoun, A. Khajeh, N. Dutt, A.M. Eltawil, and F.J. Kurdahi. E < MC2: Less Energy through Multi-Copy Cache. In Proc. of Int. Conf. on Compilers, Architectures and Synthesis for Embedded Systems (CASES), pages 237-246, 2010.

[16] A. Sasan, H. Homayoun, et al., "A fault tolerant cache architecture for sub 500mV operation: resizable data composer cache (RDC-cache)," in Proc. CASES 2009.

[17] H. Homayoun, S. Pasricha, M.A. Makhzan, A. Veidenbaum, Improving performance and reducing energy-delay with adaptive resource resizing for out-of-order embedded processors. In: Conference on Languages, Compilers and Tools for Embedded Systems (2008).

[18] S. R. Nassif "Modeling and Analysis of manufacturing variation" in Proc. CICC, 2001.

[19] Wilkerson C. et. al.. "Trading off cache Capacity for Reliability to Enable Low Voltage Operation." ISCA 2008.

[20] H. Homayoun, M. Makhzan, A. Veidenbaum, "ZZ-HVS: Zig-Zag Horizontal and Vertical Sleep Transistor Sharing to Reduce Leakage Power in On Chip SRAM Peripheral Circuits", In Proceedings of IEEE International Conference on Computer Design, ICCD 2008, U.S.A.

[21] http://download.intel.com/design/processor/datashts/320032.pdf.