Inquisitive Defect Cache: A Means of Combating Manufacturing Induced Process Variation

Journal:	*IEEE Transactions on Very Large Scale Integration Systems*
Manuscript ID:	TVLSI-00193-2009.R1
Manuscript Type:	Regular Paper
Date Submitted by the Author:	07-Oct-2009
Complete List of Authors:	sasan, avesta; uci, eecs HOMAYOUN, HOUMAN; uci, eecs Eltawil, Ahmed; University of California, Irvine, Electrical Engineering and Computer Science Kurdahi, Fadi; UC Irvine, ECE Dept.
Key Words:	LOW POWER, PROCESS VARIATION, MEMORY, CACHES, FAULT TOLERANCE

Inquisitive Defect Cache: A Means of Combating Manufacturing Induced Process Variation

Avesta Sasan (Mohammad A Makhzan), Houman Homayoun, Ahmed Eltawil, Fadi Kurdahi,
{mmakhzan,hhomayou, aeltawil,kurdahi}@uci.edu

Abstract-This paper proposes a new fault tolerant cache organization capable of dynamically mapping the in-use defective locations in a processor cache to an auxiliary parallel memory, creating a defect-free view of the cache for the processor. While voltage scaling has a super-linear effect on reducing power, it exponentially increases the defect rate in memory. The ability of the proposed cache organization to tolerate a large number of defects makes it a perfect candidate for voltage-scalable architectures, especially in smaller geometries where Manufacturing Induced Process Variation (MIPV) is expected to rapidly increase. The introduced fault tolerant architecture consumes little energy area overhead, but enables the system to operate correctly and boosts the system performance close to a defect-free system. For a 16KB L1 cache using an Inquisitive Defect Cache (IDC) of size 1KB, power savings of over 40% is reported on standard benchmarks while the performance degradation is maintained below 1%

Index Terms – Process variation, Low Power, SRAM, Cache, Fault Tolerance, voltage scaling, power efficient, technology scaling, manufacturing defects.

I. Introduction

The tradition of using minimum sized transistors to create compact SRAM structures, coupled with the analog nature of memory read/write operations, results in a much larger defect density in memories as compared to logic. In addition, it is shown [1][2][17] and will further be discussed in section III that voltage scaling exponentially increases the memory cell failure probability. This implies that caches, which are the largest memory structures in the processor, abruptly limit the processor lower limit of supply voltage by introducing an exponential relation between their bit failure rate and the system supplied voltage. Since the failure probability under voltage scaling is exponential, fault tolerant techniques, such as row and column redundancy, that linearly increase the fault tolerance limit of the cache will fall short for significant reductions of the supply voltage (V_{cc}^{mim}) due to their limited fault coverage. This suggests that a fault tolerant mechanism, to be used in voltage scaled caches, should be able to tolerate very large number of defects. The increase in fault coverage however should be achieved in a fashion such that 1) the impact of fault handling mechanism on delay and/or performance of system is acceptable, 2) the total power consumption of the system including the new fault handling mechanism is lower than that of other simpler fault handling mechanism that operate at higher voltages handling lower number of defects with lower power/overhead.

In this paper, we explore the organization of a novel fault tolerant cache architecture which is capable of providing fault coverage for caches suffering from very high defect rate. We first introduced this organization in [3] for direct mapped caches with single word fetch per cache access. In this work, we further explore the usage of the IDC to associative caches with multiword fetch groups. We also introduce a mathematical upper bound on defect coverage of the proposed architecture relating the area and power of the proposed architecture to achievable V_{cc}^{mim}. In [3] we investigated the fault coverage and voltage scalability of the system when a Fixed Frequency Voltage Scaling (FFVS) policy for scaling the cache supplied voltage is used. In this work, we extend our study and results examining the utilization of the proposed architecture for both Voltage Frequency Scaling (VFS) and FFVS policies. In [3] for obtaining the power and leakage figures we utilized CACTI [4] power and leakage estimations, however in this work, in order to improve the accuracy of our results, the leakage and power figures are replaced with those obtained from post layout simulation of the cache layout. Furthermore, the simulation environment (DINERO IV) presented in [3] has been replaced with SimpleScalar [5] to allow simulation of the architecture for well known SPEC2000 bench marks.

The paper is organized as follows: Section II builds the necessary background and highlights the motivation behind this work. Section III briefly explores the effect of supply voltage scaling on the number of MIPV defects, explaining how the access/write time distribution of the cache changes with changes in supply voltage and/or frequency. Section IV, explores the relation between defining memory access time and MIPV failure rate, explaining the relationship between voltage scaling, frequency scaling, performance, system power and energy consumption. Section V introduces our proposed architecture which guarantees the correct operation of the cache and at the same time creates a virtually defect-free view of the cache for the processor. Section VI presents an energy model to quantify the energy savings, while section VII presents a mathematical bound on IDC coverage. In section VIII, we validate our proposed concept trough SimpleScalar simulation, illustrating how it could reduce overall system energy and power consumption by tolerating a larger numbers of defects and therefore allowing better performance at nominal voltages or alternatively the same performance at lower voltages. Finally, the paper is concluded in section IX.

II. Prior Work

Cache is a high speed SRAM memory used to bridge the speed gap between the processor and slower Lower Level Memories (LLM). For accessing LLM, one pays the price of longer access time and large power consumption. By reducing the number of accesses to LLM, cache decreases the access time and improves the overall dynamic power consumption. It is the temporal and spatial locality of instruction and data that allows the architect to use caches for holding data and instructions closer to the CPU. If a cache is defective, the system could still operate correctly provided that the faulty cache block or word could be disabled. One way to achieve this is by marking a defective cache word/block with an extra bit that can be added to the set of flag bits of that block, conditioned that the added bit is not defective. In [6] this bit was called the Fault Tolerance bit (*FT-bit)*. The set of FT-bits for memory words or blocks is called the *defect map*. This defect map is used to turn a cache line off in case it is faulty. Turning a cache line off in an associative cache reduces the degree of associatively by one. In a direct mapped cache, on the other hand, every access to a disabled line will result in a miss. This scheme of disabling faulty blocks is done even when the cache is protected by *single-error correcting, double-error detecting* (SEC-DED) codes [7].

Replacement techniques, such as extra rows and columns, are also used in caches for yield enhancement [20] and for tolerating lifetime failures[8][9][10][11]. With replacement techniques, there is no performance loss in caches with faults. However, the number of extra resources limits the number of faults that can be tolerated using these techniques. Another form of redundancy is the use of extra bits per word to store an error correcting code. Sohi[8] investigated the application of a Single Error Correcting and Double Error Detecting (SEC-DED) Hamming code in an on-chip cache memory and found out that it degrades the overall memory access time significantly.

The work in [12] suggested resizable caches. In this technique it is assumed that in a cache layout, two or more blocks are laid in one row, therefore the column decoders are altered to choose another block in the same row if the original block is defective. In a technique called PADded caches [6] the decoder is modified to allow remapping of the defective blocks to other locations of the cache without destroying the temporal locality of data or instruction. A recent paper from Intel's microprocessor technology lab [13] suggested the use of fault tolerant mechanisms trading off the cache capacity and associatively for fault tolerance. The proposed approaches while reducing the frequency allows scaling the voltage from a nominal 900mv down to 500mv in a 65nm technology. The cache size is reduced to 75% or 50% depending on the mechanism that is used. In this scheme, data that is mapped to defective location is reallocated to healthy locations. The delay overhead, and the smaller cache size suggests that this scheme could not be used when high performance is desired, since, not only do we have to reduce the frequency, but also, due to smaller cache size, the number of cache misses will increase.

In this paper, a new architecture is presented specifically addressing power and yield improvement via fault tolerance. The ability to tolerate process and operational condition induced faults allows aggressive voltage reduction even in high performance mode (Fixed Frequency), which in turns leads to reduced power consumption

III. Effect of Voltage Scaling on Memory

A. *Classification of Memory Errors*

Memory cell failures are generally divided into three distinct categories:

1) Fixed Errors or Manufacturing defects, which are un-accounted shorts and opens in the memory cell and its supporting logic resulting in erroneous operation.
2) Transient Errors which are temporary memory malfunctions as a result of a transient situation such as an alpha particle attack.
3) Operating condition dependent in the result of manufacturing process variation in device

physical parameters. This group of failures is not always present however with change in the memory operation conditions including: Process, Voltage & Temperature (PVT) their number is changed. Symptoms of these failures are change in cell access times, or unstable read/write operations in some locations upon change in PVT.

Significant contributors to the rapid increase in the number of process variation induced defects include line roughness, oxide thickness variations, as well as Random Dopant Fluctuations (RDF) which results in mismatch between adjacent transistors threshold voltage (V_{th}) [1]. We will refer to such variation as Manufacturing Induced Local Process Variation (MILPV). When applied to memory cells, the V_{th} variation results in large variation in access and write time to the memory cells. The dependence of V_{th} on temperature makes the write/access time also sensitive to die temperature. Furthermore, since the transistor speed is a strong non-linear function of V_{dd}-V_{th} , the access/read time is also strongly and non-linearly dependent on the supply voltage.

Table 1: Change in the mean and access time

Voltage	μ	σ
0.9	43.77	7.504
0.8	65.75	13.873
0.7	91.6	19.987
0.6	136.9	26.35
0.5	197.54	37.038

B. Modeling Memory Access Time Under Process Variation

We setup an experiment to model the changes in the access and write time distribution of a memory when its transistor devices are affected by MILPV. We lumped & modeled the MILPV as an independent Gaussian distribution characterizing the V_{th} fluctuations of each transistor device[14]**.** The circuit under test is a standard six transistor SRAM memory bit cell. The SPICE models used for the simulation were obtained from the Predictive Technology Model (PTM) [15] website in 32nm.

Due to the random and uniform distribution of defects, it was expected and verified by Monte-Carlo simulation that the access/write time to the cache follows a "Gaussian like" distribution. However as the supply voltage of the memory cell changes, the characteristic of the access/write distributions also varies. We repeated 10K Monte-Carlo Simulation for each voltage point. By fitting the obtained iteration points at each voltage into a Gaussian distribution we obtained the associated mean and standard deviation. This data is presented in Table 1. It was observed that with changing the supply voltage not only does the mean access time change, but also the standard deviation of the fitted distribution.

C. Defining System Access Time, Safety Margin (SM) and its Implications on Memory Cell Failure Rate

The conventional model for defining an access time at a voltage point when the access distribution is known is to choose an access time large enough that Cumulative Distribution Probability of the distribution tail exceeding the defined access time is very small. This is illustrated in Figure 1. This probability is determined by the designer depending on the expected yield and amount of dedicated redundancy available. The gap between mean access/write time and defined cycle time is referred to as 'Safety Margin' (SM). Choosing a large SM will result in higher yield however it adversely degrades the system's performance.

As the supply voltage is down scaled, both the mean (μ) and standard deviation (σ) of the access/write time increase. It is however still possible to define the cycle time at each voltage such that the probability of failure stays constant, but moving to lower voltages increases the mean and standard deviation quickly and trying to maintain the same probability of failure results in extremely long access time and therefore extremely poor performance. On the other hand, if the cycle time is chosen such that failure probability is traded for performance, we will be expecting a higher failure rate which, if no fault tolerance mechanism is in place, means a lower production yield. Depending on how the cycle time is chosen, the mapping of the probability of failure to the supply voltage changes. The Access time at each voltage point could be defined using the following equation:

$$T_{Access}(V_i) = a_{vi}.(\mu_{Vi}) + b_{vi}.(\sigma_{Vi}) + c_{vi} \quad (1)$$

In which a_{vi} and b_{vi} are scale factors, c_{vi} is a constant, and μ_{Vi} and σ_{Vi} are the mean and standard deviation of the access distribution at scaled voltage V_i. From Equation (1), SM is defined as:

$$SM(V_i) = (a_{vi} - 1).(\mu_{Vi}) + b_{vi}.(\sigma_{Vi}) + c_{vi} \quad (2)$$

Many different voltage scaling policies could be defined by choosing the values for a_{vi}, b_{vi} and c_{vi}. An extreme approach is Fixed Frequency Voltage Scaling (FFVS) in which a_{vi} and b_{vi} are chosen "0", and c_{vi} is a constant value throughout the voltage scaling range. This makes SM a decreasing function of V_i.

Reduced SM means less tolerance to variation. As the gap between defined cycle time and mean access time decreases, a larger tail of access distribution will be mapped outside of the defined cycle time. This results in exponential increase in probability of cell failure.

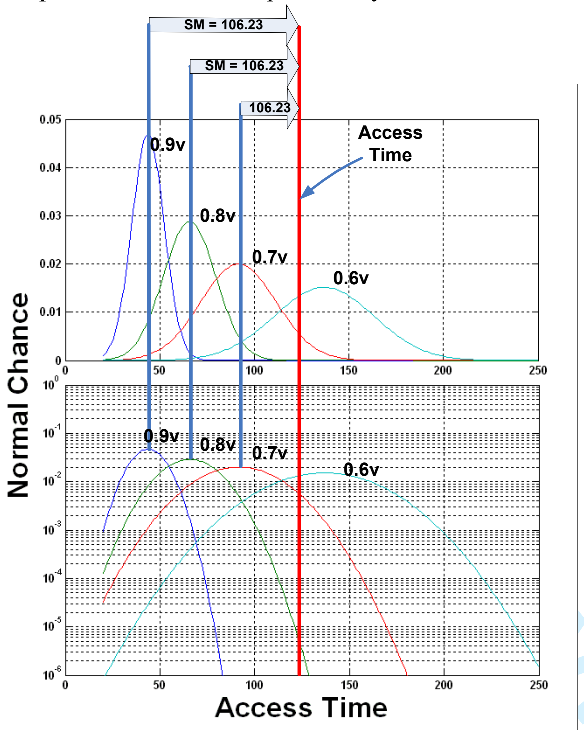

Figure 1: Voltage Scaling and change in the mean, and standard deviation of access time distribution and change in Safety margin for an FFVS policy with cycle time of 125ps

Simulation results of 10^7 Monte Carlo iterations for obtaining the failure probability across different voltages when the FFVS is used is illustrated in Figure 2. At each voltage point the access time to a 6T cell in an array of memory cells under process variation is obtained. If the resulting access time or write time exceeds the defined cycle time, or the content of the cell is flipped during a read operation the memory cell is considered weak or defective contributing to the cell failure probability. The failure probability in all presented cases (except when excessively large access times are chosen) grows exponentially. Clearly, curves associated with smaller access times experience higher failure rates.

Maintaining the same cycle time while scaling Vdd (FFVS) allows the system to perform as fast as it does under nominal voltage. While voltage scaling reduces the power consumption, the expected system yield is adversely affected. Having a Fault Tolerance Mechanism (FTM) in place could improve the yield of such system. Designing a FTM in a voltage scalable environment includes determining the expected yield, determining the maximum tolerance of the FTM and finally obtaining a lower bound on voltage scaling at which the failure rate exceeds the FTM tolerance rate. This model is of interest since it allows full speed operation at lower a voltage.

Changing the frequency along with voltage allows the system to migrate and operate at even lower voltages. In the Voltage Frequency Scaling (VFS) policy a_{vi}, b_{vi} and c_{vi} are redefined at each voltage. This allows the designer to choose larger cycle time at lower voltages trading failure rate for performance.

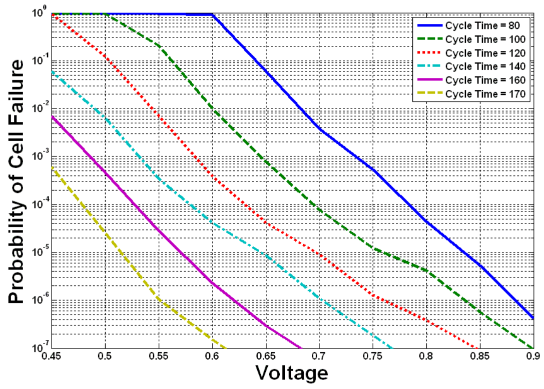

Figure 2: Probability of cell failure for FFVS when different cycle times are chosen

Figure 3 illustrates the probability of cell failure for a system with VFS in place. The graph shows the failure rates for a few different choices of a_{vi}, b_{vi} and c_{vi}. Table 2 summarizes the SM associated with the different choices of these parameters across different voltages. Information in this table could be easily related to the failure rate curves shown in Figure 3 where choosing a larger or smaller safety SM implies a slower or faster increase in the failure rates respectively.

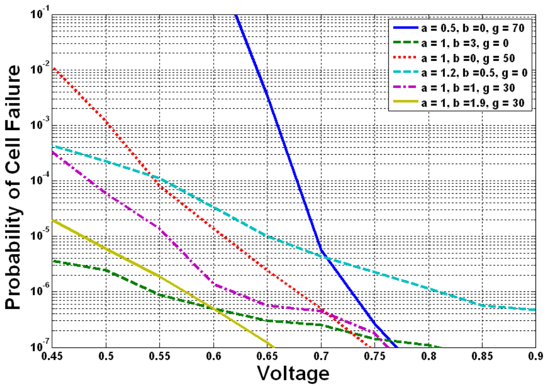

Figure 3: Probability of cell failure for VFS for different choices of a, b and c in equation 1

As illustrated, VFS could be done in numerous ways. The defined frequency of operation at each voltage point relates the probability of failure to that specific voltage level. By choosing a larger access time (and therefore larger safety margin) at lower voltages, the system's ability to tolerate process variation is increased; however the increased system cycle time deteriorates performance. The implication on power consumption is twofold. On one hand, the lower voltage results in lower dynamic and leakage power consumption. On the other hand, the lower performance and the resulting larger execution time imply that the circuits run for a longer time, which increases the total energy consumption. The appropriate lower limit on the voltage should be chosen by considering the maximum tolerable defect rate, the desired performance and power consumption.

Table 2: Mean access time for different choices of SM parameters across scaled voltages.

	0.5	1	1	1	1	1	a
	0	3	0	0.5	1	1.9	b
V	70	0	50	0	30	30	c
0.9	48.1	22.5	50	12.5	37.5	44.2	
0.85	43.7	30.0	50	15.5	40.0	49.1	
0.8	37.1	41.3	50	20.0	43.8	56.2	
0.75	30.8	52.8	50	24.5	47.6	63.4	
0.7	24.2	59.9	50	28.3	49.9	67.9	
0.65	14.3	67.2	50	33.5	52.4	72.5	
0.6	1.6	79.0	50	40.6	56.3	80.0	
0.55	-12.1	96.4	50	48.9	62.1	91.0	
0.5	-28.8	111.1	50	58.0	67.0	100.0	
0.45	-58.3	132.9	50	73.5	74.3	114.1	

IV. Effect of Voltage Scaling on Memory Logic

In this section, we focus on a three stage pipeline cache as a case study, but note that the same arguments could be applied to wave pipelined and non-pipelined caches as well. We specifically choose the 3 stage pipeline cache to show some of the design considerations when a system is designed for FFVS or VFS policy*[1].

In a three stage pipeline cache the cache operation is divided into; a) Decoding in the first stage, b) Memory cell access in the second stage, which is considered from activation of the wordline, to the analog operation of reading the target memory cells into bitlines and ends with sense amplifier activation and, finally c) Sensing, muxing and driving the output signals in the third stage. In this configuration the second stage (reading the memory cells) is the longest pipeline stage. This stage is analog and cannot be pipelined into further stages [1]. Figure 4 illustrates the timing of a three stage pipelined cache working at nominal V_{dd}. When pipelined, every stage in the cache has one cycle to complete its operation. The cycle length is determined by the delay of the longest stage in pipeline.

*[1] Although FFVS could be considered as a special case of VFS however the design implications-- especially in pipeline structures-- explain why FFVS is considered separately.

In a FFVS system the second stage delay stays constant throughout the voltage scaling and the shift in the distribution of the access/write time is translated to higher failure rates. One determinant for the lowest possible voltage V_{cc}^{Min} is the tolerable failure rate. Another consideration for such system is in design of other stages of the pipeline (first and third stage in this case). These stages are purely logic stages. Reducing the voltage increases the logic propagation time and therefore reduces the operation speed of these stages. The delay of these stages, however, cannot exceed the delay of the second stage at lower voltages. Therefore in FFVS system they are designed such that all the logic operations still complete correctly at the lowest voltage. This means potentially using larger and faster transistors in the design.

On the other hand, if the VFS policy is used; as the voltage is lowered, the second stage delay (Δ) increases. Since all the stages of the pipeline should take equal time the delay of the second stage is multiplied by the number of stages in the pipeline, increasing the access time by $N.\Delta$ and cycle time by Δ. In such a case the delay slack in the first and third pipeline stages allows the usage of smaller transistor in those stages to achieve lower power consumption.

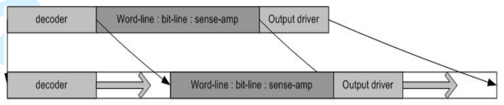
decoder
Word-line : bit-line : sense-amp
Output driver
decoder
Word-line : bit-line : sense-amp
Output driver

Figure 4: Converting the access time of a non pipelined cache to a pipelined cache

The delay of the logic [21] in the first and third stages is related to the supply voltage by the following equation:

$$Delay = \frac{V_{dd}}{(V_{dd} - V_{th})^{1.3}} \qquad (3)$$

Table 3 depicts the delay of various stages in a 16KB cache. The delays are obtained by simulating a post-layout realization of a 16KB cache whose parameters are described in the same table. Date presented in table 3 includes the delay of the decoder, memory cell, and the output logic for both tag and data. In this paper, we assume that all memory components are controlled by a single variable supply voltage. This assumption is made for the sake of simplicity and implementation purposes. In case of latches, we should make an exception and consider the latches to always work at the cache's nominal V_{dd}. The first and third stages have a lower limit on how much their voltage could be reduced before the stage delay exceeds the second stage delay (which defines the cache cycle time). In the proposed cache, when reducing the voltage, the tag decoding stage delay reaches the cycle time limit before other logic stages and since all stages are connected to a single supply voltage, this voltage defines the lower limit of supply scaling (V_{cc}^{mim}).

Table 3: Cache parameters for a 16KB cache obtained from a post layout simulation in 32nm.

Data stage delays(ps)		Tag stage delays(ns)		
decode	13	decode	17	
total	13	Total	17	1
Word-line	16	Word-line	14	
Bit-line	22	Bit-line	21	
sense amp	7	Sense amp	6	
total	45	Total	41	2
output driver	16	compare	15	
		valid signal	6	
total	16	Total	21	3

Cache Parameters:

Number of banks	**1**
Total Cache Size (bytes)	**16386**
Number of sets per bank	**1024**
Associativity	**4**
Block size(bytes)	**16**

V. Proposed Architecture

Figure 5 illustrates our purposed architecture. A cache in this model is augmented with two auxiliary blocks. The Bit Lock Block (BLB) is a defect map for the cache and the IDC is a small auxiliary cache for the purpose of defect tolerance. Both of these auxiliary structures are kept at high supply voltage when the voltage of the main cache is scaled. This is to guaranty reliable storage of defect information in the defect map. The size of these two structures compared to cache is much smaller and it will be shown that their augmented power consumption is also much smaller than that saved by allowing more aggressive voltage scaling in the cache. We describe the functionality, structure and relation of these components in the following sections.

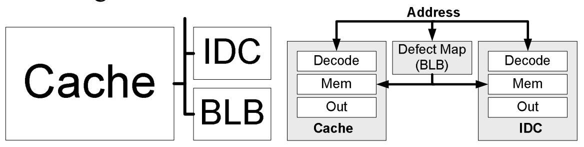

Figure 5: Proposed architecture

A. Bit Lock Block (BLB) defect map

The number of words that are read from a cache at each read cycle is referred to as a Fetch Group (FG). In this architecture, to create a defect map, we use one bit for each FG in cache, which is stored in the BLB. Unlike some previous implementations of defect map for caches, where a portion of the tag bits were used for keeping the defect map, we separated the defect structure from the tag section to allow scaling the voltage of the tag bits along with memory bits while the defect information is kept at a high supply voltage. This guaranties a smaller structure for defect map and faster access to the defect information.

B. Generating and Updating the BLB

As part of the operation of the system, we need to have a mechanism for generating the BLB and another one for updating it. Generating the BLB could be accomplished at manufacturing time, however testing for multiple vectors significantly increases the chip cost. Another approach is to generate the defect map at boot time. In this method, at manufacturing time, the memory is only tested for the manufacturing defects at the nominal voltage process corners, fixing such errors using the available redundancy and leaving the process variation induced defects to the combination of the FTM and Built In Self Test (BIST) unit. At run time BIST tests the memory arrays at lower voltages and write the defect map into BLB. BIST might generate a different BLB for operation at different PVT corners. Defect maps can be saved by the system in non volatile memory to avoid running the BIST in the future or can be re-run on demand.

To maintain current information in the BLB that takes into consideration operation environment changes (e.g. temperate), we need to consider a capability for dynamically updating the BLB. This is achieved in one or a combination of the following ways: (1) Usage of parity bits: Every time that a new error is discovered, its location is registered in a very small memory (in our case 16 or 32 entry). Since the newly discovered defect could be a soft error, BLB is not updated immediately. If the defective FG is detected again then it is considered a temperature induced error and is registered in BLB. Each BLB row has a dirty bit which is set if a bit in that row is updated. The presence of the dirty bit requires a write back of the BLB information when changing the voltage level. (2) Run the built in BIST engine periodically. (3) Using Dynamic Temperature Sensing (DTS) infrastructure, an approach which is already being adopted by many manufacturing companies in 65nm and technologies and below. DTS is geared towards avoiding overheating but can also be used for other purposes such as modifying the policy stacks. In this approach, the system senses the cache temperature. When a temperature increase exceeding a pre-specified amount is detected, the BIST engine is instructed to test the heated locations for temperature induced defects. The BLB is updated and the dirty bit is set. Since temperature is a very slow changing variable such BIST updates will not run frequently. In addition, as the chip ages the BLB coverage increases and less

BLB updates are required. (4) Finally, another approach which could be combined with previous methods is an adaptive technique which decides on the voltage level by monitoring the number of dynamically generated defects. In this adaptive approach, the voltage will be increased to mitigate/mask the high rate of dynamically generated errors. However, the earlier power savings will be diminished.

Every time the voltage changes, the incremental defect maps are loaded into BLB. If there are any dirty bits a write back is needed before loading the new defect map. The BLB itself is very small compared to the cache and therefore it can operate much faster than the cache. Furthermore, the BLB voltage is not scaled, therefore its delay never exceeds the cache cycle time.

Although IDC is designed for dealing with process variation defects, it could also be used to deal with manufacturing defects in upper level memories (caches). In other words, manufacturing defects could be considered as process variation defects that are present in all corners of the PVT space. With this in mind if at the manufacturing the number of available redundancy is less than number of manufacturing defects, conditioned that their number is smaller than available space in IDC and they are distributed in FGs far apart, they could be masked by the proposed mechanism.

C. *Inquisitive Defect Cache memory*

The Inquisitive Defect Cache (IDC) is a small associative cache which temporarily holds FGs mapped to defective locations in its associated cache while these FGs are within the Window of Execution (WoE). The conceptual view of an IDC enabled cache structure is illustrated in Figure 6.

The size of IDC cache could be much smaller than the total number of defective FGs in a cache. In a sense, the IDC could be considered as a group of redundant rows with the main difference being that mapping to redundancy is fixed and one time, whereas mapping to the IDC is dynamic and changes with changes in the addressing behavior of the program. Furthermore, in the IDC case, the defective FGs within the Window of the Execution (WoE) of the program in the cache are mapped to IDC allowing much higher coverage than that of a redundancy scheme.

In Figure 6, IDC, Cache and BLB represent the Inquisitive Defect Cache, Cache and Defect Map (Bit Lock Block) respectively. The WoE and "Int" are conceptual structures (do not exist but drawn to ease the understanding of IDC structure and its operation) representing the Window of Execution of the program in the cache and Intersection of the WoE and BLB respectively. The BLB marks the defective FGs in the cache. The WoE marks the in-use section of the cache (FGs that the processor has recently visited). The "Int" which is the intersection of BLB and WoE is then the defective FGs in the WoE which should be mapped to IDC. In Figure 6 the IDC and cache have associativity of 2 and 4, respectively.

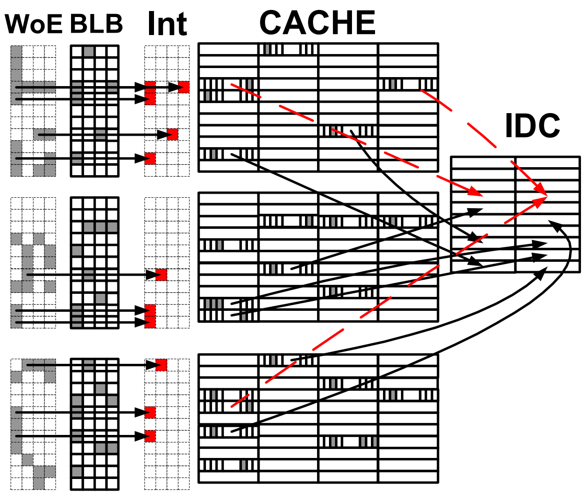

Figure 6: In the first pass defective FGs in window of execution are mapped to IDC.

If the IDC size and associatively are chosen properly, the WoE of a process in the cache (after its first pass) should experience very few defective/disabled words and could be virtually viewed as a defect free segment in the cache by the processor.

If a FG is defective, its information could only be found in the IDC since every read/write access to defective words is mapped to the IDC. The BLB should be designed such that its access time is smaller than the cache cycle time. Considering the small size of the Defect Map when one bit per FG is chosen, and also the fact that the Defect Map voltage will not be scaled, makes it fairly easy to meet this design constraint. On each access all three components (BLB, IDC and Cache) are accessed simultaneously. However immediately after BLB feedback the operation on either cache or IDC for the next cycle is gated to save power. The following section analyzes the access scenarios and derives models for energy consumption for each scenario.

VI. Energy Models

A. *Cache access scenarios*

An access to the cache, as illustrated in Figure 7, initiates parallel access to cache, IDC & BLB. In this model however after the first cycle, based on defect map feedback, access to either cache or IDC is gated. The dynamic energy consumption when IDC or cache is gated is presented in equations (4) and (5).

If for some design reason the defect map access, could not fit in one cycle, a buffer could be added to act as a cache for the BLB. The buffer consists of a set of flip flops and each access to the BLB is serialized with access to the BLB buffer. In this case, the access could either be gated at the end of Cycle1 (if information is in the buffer) or Cycle2 (if the information in not in buffer and BLB should be accessed), while stopping further access to either the cache or IDC [3]. The timing of this sequence of operations is given in Figure 8. Due to temporal and spatial locality of access, the defect map query is usually resolved by the buffer, saving time and energy even with a BLB that takes longer than one cycle to access. However, in this paper, we build our system based on a BLB capable of providing the defect information in one cycle. The reader is referred to [3] for further detsils on scenarios considering the BLB buffer.

$$E = (E_{BLB} + E_{Didc}) + (E_{Rc}) * \frac{V_{dd}^{2}}{V_{dd0}^{2}} \qquad (4)$$

$$E = (E_{BLB} + E_{IDC}) + (E_{Dcache}) * \frac{V_{dd}^{2}}{V_{dd0}^{2}} \qquad (5)$$

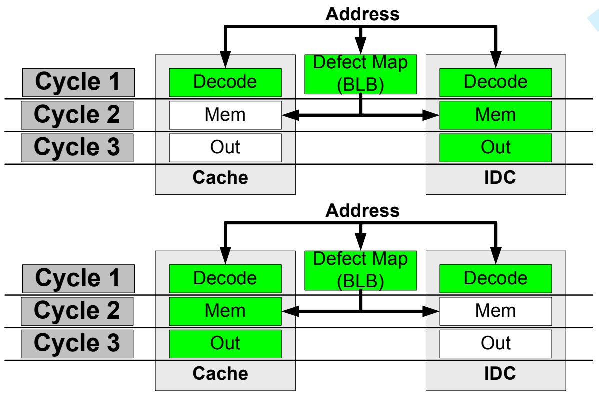

Figure 7: Cache access scenarios (top): defect map refer to IDC, (bottom): defect map refer to cache.

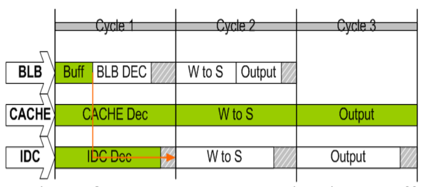

Figure 8: cache access scenario with a Buffer

In Section VIII, we report the simulation results of the proposed architecture in 32 nm technology. We use SPEC2000 bench marks to evaluate the system performance under different input conditions. We evaluated the tolerance of IDC at different voltage points under different voltage scaling policies. For each simulation point, a probability of failure per memory cell is obtained. Using that probability of failure we have created 250 defective caches and simulated the behavior of the proposed architecture. In order to quantify the energy savings, we considered both dynamic and leakage energy consumption. In the following, we illustrate how dynamic and leakage power consumption are obtained.

B. Dynamic cache energy model

As a program shifts its WoE to a new location in cache, it may access defective locations that are not currently in the IDC. At this point, the cache will declare a miss and data is obtained from lower levels of memory hierarchy and placed in IDC. For future reference, we define such a miss (a miss due to access to a defective location in the cache which is not covered in IDC) as a "*soft miss*". A soft miss could also occur if the number of defective FGs that are mapped to one physical row in the IDC exceed the associatively of the IDC (this case is illustrated using dashed lines in Figure 6).

Dynamic power is affected by the miss rate. As the miss rate increases, the dynamic power consumption increases because of the increased access to next level caches and main memory. At the same time a high miss rate could cause the processor to stall, increasing the execution time. This in turn means that the processor and memory would run for a longer time, increasing the total energy consumption. Considering that in smaller geometries leakage energy is comparable and even predicted to be dominant compared to dynamic energy, a considerable energy overhead is expected.

Consider a cache access scenario that results in a miss: The address is sent to BLB, IDC and Cache. All three components start decoding the address at the same time. Following one of the two cases explained in section VI, a miss will occur, at which point, the lower level memory is accessed for data. When data is ready in the lower level cache, it is transferred to upper level either by writing to IDC and/or Cache. When data is updated in the upper level memory an interrupt is sent to the processor and the cache request is reinitiated. At this time the BLB, IDC and Cache are accessed again in parallel. Similar to the initial cache access, one of the 2 access scenarios explained in section VI will now find the data either in the cache or the IDC. To quantify the energy expenditure, all terms used are detailed in Table 4, where the total energy per miss can be calculated as follows:

$$E_{miss} = E_{blb} + AorB(E_{Rmc} \times \frac{V_{dd}{}^{2}}{V_{dd0}{}^{2}} + E_{Didc}, E_{Rmidc}$$
$$+ E_{Dc} \times \frac{V_{dd}{}^{2}}{V_{dd0}{}^{2}}) + E_{mem} + E_{blb} +$$
$$AorB(E_{Wc} \times \frac{V_{dd}{}^{2}}{V_{dd0}{}^{2}}, E_{Widc}) +$$
$$AorB(E_{Rc} + \times \frac{V_{dd}{}^{2}}{V_{dd0}{}^{2}} E_{Didc}, E_{Ridc} + E_{Dc} \times \frac{V_{dd}{}^{2}}{V_{dd0}{}^{2}}) \qquad (6)$$

The $AorB$ function dictates that only one of its fields will be involved in the energy equation.

$$AorB(E_{Wc} \times \frac{V_{dd}{}^{2}}{V_{dd0}{}^{2}}, E_{Widc}) \qquad (7)$$

For example, AorB in Equation 7 indicates that the energy that has occurred during the write is either the write energy to the IDC or scaled write energy to the cache (scaled to factor the voltage scaling effect).

Table4: Terms and variables used

E_{blb} =	**Energy per BLB read.**
E_{mem} =	**Energy of access to the lower level memories**
E_{Rc} =	**Energy consumed to read cache**
E_{Wc} =	**Energy of a write to cache**
E_{Ridc} =	**Energy to read IDC**
E_{Widc} =	**Energy to write to IDC**
E_{Rmc} =	**Energy to read cache when miss (no output driver activation)**
E_{Rmidc} =	**Energy to read IDC when miss (no output driver activation)**
E_{Dc} =	**Energy to decode cache**
E_{Didc} =	**Energy to decode IDC**
$AorB(A,B)$ =	**A If case A happens, B if case B happens**
V_{dd} =	**Scaled memory supply voltage**
V_{dd0} =	**Nominal memory supply voltage**
L =	**Size of the Window of Execution**
A =	**Associability in the WoE**
S =	**Associability of the IDC cache**
D =	**Size of the IDC cache**
P_f	**Probability of Cell failure**
W =	**Number of Words in each fetch group**
T =	**Number of bits in each word**
P_{clean_access} =	**Probability of clean access**

C. *Leakage power analysis*

As discussed previously, leakage energy consumption is also affected by the miss rate. A miss will/could potentially increase the program execution time and therefore the circuit would leak for a longer period of time. Memory leakage power consumption is exponentially related to the memory supply voltage. At process technologies of 65nm and below, the dominant leakage components is sub-threshold leakage. The sub-threshold leakage in MOSFET transistors relates to voltage by:

$$I_{leakage} = \mu_0.C_{ox}.\frac{W}{L}.e^{b(V_{dd}-V_{dd0})}.v_t^2.(1-e^{\frac{-V_{ds}}{v_t}}).e^{\frac{-\Delta V_{th}}{n.v_t}} \qquad (8)$$

In this equation, $v_t = KT/q$ is the thermal voltage, W is the device width, V_{ds} is the drain to source voltage, n is the drain induced barrier lowering (DIBL) coefficient, and $\mu_0.C_{ox}$ are constants that do not depend on temperature or voltage. The constant term b in the equation is a technology dependent coefficient which is obtained empirically. CACTI4.2 [4] assumes $b = 1.7$ in 65 nm technology. For 32nm, we ran SPICE simulations of a 6T memory cell using *W/L* ratio of 2, and choosing the widths of Pre-charge and Write NMOS transistors as 16X the size of memory cell NMOS width. Using linear regression on the simulation results, we obtained an estimate of 2.1 for b in 45 nm technology and 3.4 for 32nm technology. In this equation, everything is constant except W, V_{dd}, T and V_{th}. Therefore the equation could be written as:

$$I_{leakage} = W.I_l(T, V_{dd}, V_{th}) \qquad (9)$$

Using PTM V1.0 [15] and considering the cache to be operating at 360° K, the total leakage of our proposed cache architecture at different voltage levels for 32 and 45 nm technologies is compared to the leakage of a traditional cache architecture and is illustrated in Figure 9.

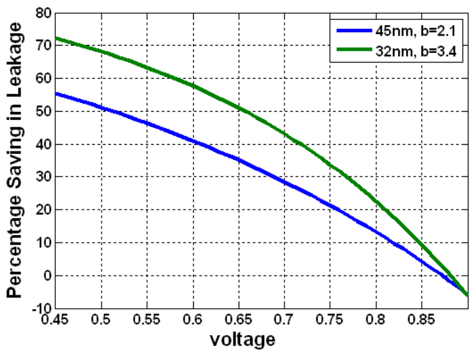

Figure 9: Percentage saving in the leakage power in proposed architecture compared with the traditional cache architecture

Leakage power includes the extra leakage introduced by BLB and IDC cache. At nominal voltage the leakage saving compared to a traditional cache (with no IDC and BLB) is negative because of the additional leakage overhead from IDC cache and BLB*[2]. However, at lower voltages, (as will be shown in the results section (section VIII)) saving in the leakage of cache far outweighs the extra leakage expensed. Notice that only the voltage in the main cache is scaled while the BLB and IDC are kept at their nominal voltages. The leakage savings in the 32nm grows more rapidly than the 45nm technology since the *b* factor in Equation (8) in 32nm technology is higher than in 45nm technology. (Both technologies are simulated using identical threshold voltages).

VII. MATHEMATICAL BOUND ON IDC COVERAGE.

In this section, we develop the upper and lower bounds on the fault tolerance capabilities of the IDC. In section VIII, it will be illustrated that some benchmarks utilize the IDC better than others. The amount of locality, the size of the WoE, and the sequentiality of the program in execution all contribute to better or worse utilization of the IDC. Considering uniform distribution of MILPV defects in the cache, programs with smaller windows of execution, could better utilize the IDC cache structure. The smaller the WoE, the smaller the expected number of defects in the WoE, which improves the chances that the faults in that window are covered. In addition, the higher the locality of the information, the higher the chances that a previously mapped fault to IDC is used before it is overwritten. Finally, for sequential code (codes where the addressing sequence of their program binary within the WoE are sequential (as opposed to hyper branched)) there is uniform chance of mapping conflict across all rows in IDC improving the overall utilization of IDC. This is because as illustrated in Figure 10, only a subset of locations in the cache could be mapped to each location in IDC.

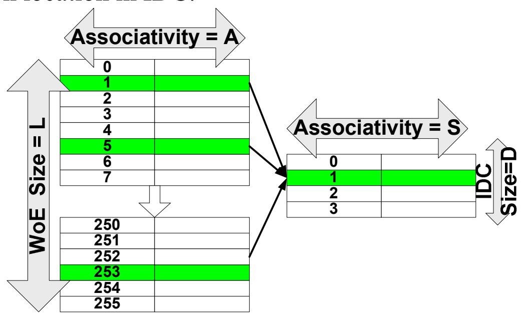

Figure 10: sequential mapping of the cache to the IDC

*[2] It is also possible to completely shut off those components when operating at nominal Vdd therefore eliminating their leakage as well using power supply gating transistors.

In order to show two extremes of fault tolerance performance, we consider two program cases: one with hyper branched addressing (almost every instruction is a branch) and the other with sequential program binary addressing. The probability of a fetch group failure (a defective FG) is:

$$P_{FG} = 1 - (1 - P_f)^{W.T} \tag{10}$$

We define a clean access as "an access to one line in the WoE which is either mapped to a defect-free FG in the cache or to a FG previously mapped to the IDC". Note that such access could still result in a miss however the miss is not due to addressing a defective FG, but rather to a failure in tag matching (i.e. the intended information is not in the cache).

Based on Figure 10, if the binary in the window of execution is highly sequential, $L.A/D$ lines in WoE (uniform distribution) could be mapped to each line. In this case, access to one line is clean if either the FG is clean, or if it is defective and is already mapped to the defective location and is not in a mapping conflict where more than 'S' FGs that need to be mapped to one line in the IDC. In another words, there is less or equal to *S-1* defects in the other $L.A/D$ lines that are mapped to the same physical address in the IDC that the current FG need to be mapped to. Thus, we have:

$$P_{clean\_access} = P(defect\_free\_FG) + (1 - P(defect\_free\_FG)).P_{\text{cov}} \tag{11}$$

In which P_{cov} is the probability that there are less that S other lines mapped to the associated line in IDC.

$$P_{\text{cov}} = P_{MFG}(0) + P_{MFG}(1) + \ldots P_{MFG}(S-1) \tag{12}$$

In which $P_{MFG}(i)$ means the probability that '*i*' other FGs in the WoE are mapped to that exact same location in IDC. Therefore:

$$P_{\text{cov}} = (1-P_{FG})^{\frac{L.A}{D}-1} + \binom{\frac{L.A}{D}}{1} .P_{FG}.(1-P_{FG})^{\frac{L.A}{D}-2} + \ldots + \binom{\frac{L.A}{D}}{S-1} .P_{FG}^{S-1}.(1-P_{FG})^{\frac{L.A}{D}-S} = \sum_{i=0}^{S-1} \binom{\frac{L.A}{D}}{i} .P_{FG}^{i}.(1-P_{FG})^{\frac{L.A}{D}-i-1} \tag{13}$$

For the case of hyper branched binary addressing of the program within the WoE, each of the lines in the window of execution is equally likely to be mapped to any line in the IDC. Therefore the probability of a clean access to one line is obtained by replacing $L.A/D$ in equation (13) by $L.A$.

The probability that every access to the WoE is clean is determined by $(P_{clean\_access})^{L}$. Therefore the probability of clean access in WoE for sequential and hyper branched addressing are:

$$P_{Cov}^{Seq} = \left[\sum_{i=0}^{S-1} \begin{pmatrix} \frac{L.A}{D} \\ i \end{pmatrix} . P_{FG}^{i} . (1 - P_{FG})^{\frac{L.A}{D} - i - 1} \right]^{L} \quad (14)$$

$$P_{Cov}^{Hyper} = \left[\sum_{i=0}^{S-1} \begin{pmatrix} L.A \\ i \end{pmatrix} . P_{FG}^{i} . (1 - P_{FG})^{L.A - i - 1} \right]^{L} \quad (15)$$

Based on this analysis it is interesting to investigate how the program behavior in terms of sequantiality could result in different utilization of the IDC. Figure 11 illustrates the probability of clean access to the WoE for two programs with identical length in WoE utilizing the same size IDC but one with sequential execution in the WoE, and the other with hyper branched binary addressing. We can clearly see that a sequential program maintains perfectly clean access even for $P_f > 10^{-4}$ (and therefore lower Vdd) while a program with hyper branched binary addressing starts to deteriorate much earlier (i.e. at higher Vdd). For practical benchmarks usually the probability of clean access is somewhere between these two extremes.

Figure 12 reveals the effect of changing the size or associatively of the IDC on the probability of clean access. The WoE in all cases is fixed and a sequential binary addressing in the WoE is considered. From Figure 12 it is concluded that increasing the associativity results in better coverage. However increasing associativity results in increased dynamic power consumption due to the increase in the number of active comparisons in each access. On the other hand, increasing associativity, potentially reduces soft misses (due to access to a defective location) and therefore total energy consumption could improve, which implies an optimal point for the associativity of the IDC. In other words, there exists a pair of (Associativity, #of Rows) for each program that will result in the lowest power consumption. However, since for most programs WoE behavior changes dynamically and different programs have different size WoEs it is not possible to find one optimal solution for all scenarios. However, the system can dynamically change the size and associativity of the IDC to target this optimal operating point.

Another interesting observation is the relation between the size of the WoE and probability of clean access, or the ability of the IDC to provide a defect free view of the WoE for the processor. Figure 13 illustrates this relationship for a sequential program.

VIII. Simulation Results

A. Simulation setup

We used SimpleScalar [5] as a system simulator to model and study how using an IDC improves MILPV tolerance and how this added tolerance is translated into energy savings. SimpleScalar is augmented with necessary extensions to model an IDC-enabled cache hierarchy. In order to obtain energy consumption metrics SimpleScalar is also enriched with bookkeeping code to calculate the energy consumption based on a per access scenario. In order to obtain a figure for dynamic energy consumption of IDC, Cache and BLB, we used post-layout energy consumption figures for 32nm technology. The simulated system configuration data is provided in Table 5.

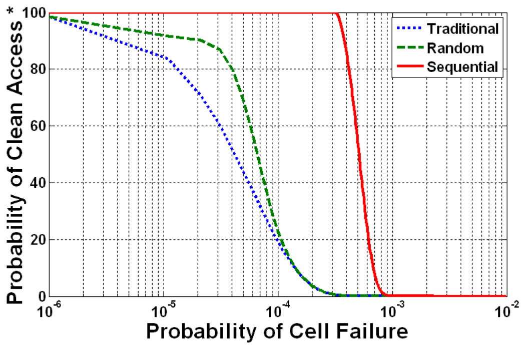

Figure 11: Probability of clean access for a complete sequential and hyper branched access to the cache when IDC structure is used, compared with Probability of clean access for the cache of the same size when IDC is not in use.

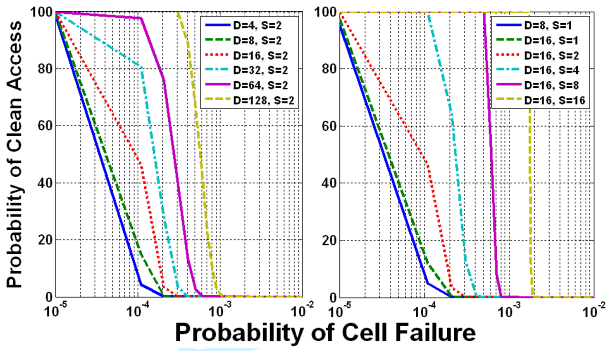

Figure 12: Comparison of Probability of clean access when changing the cache size and associativity

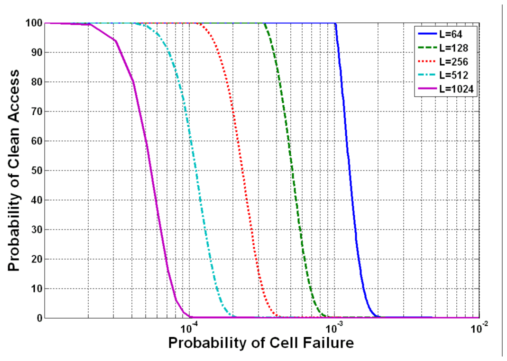

Figure 13: Probability of clean access for different sizes of WoE

Table5: SimpleScalar simulation setup

Variable	Value
Issue/decode	*8 words*
Branch prediction	*Combination*
ROB/LSQ	*128/64 entry*
L1 data cache	*32KB 3cycle pipelined, 4bank, 4 ways*
L1 inst cache	*32KB 3cycle pipelined, 4bank, 4 ways*
L2 cache	*Unified, 1M, 4way, 16cycle, 4 way*
Memory access latency	*(400 +8 per 8bytes) cycles*
L1 IDC	16 rows, 2ways
L2 IDC	64 rows,2 ways

B. *Benchmark simulation results*

The cycle time of the cache structures L1 and L2, based on Equation (1) is defined using parameters (a, b, c) = (1, 1, 30). The associated probability of cell failure curve for this simulation is drawn in Figure 3*[3]. In order to calculate the energy consumptions, based on failure probability curves, we assumed uniformly distributed defects in cache structures. After randomly fast forwarding through 1B to 2B instructions in the benchmark, we ran the benchmark for a total of 1B instructions, calculating the sum of dynamic and leakage energy consumption. The process of distribution of defects and the benchmark simulation is repeated 250 times to lessen the chances of accidental over or under utilization. For our case study, we consider the total energy consumption in L1 data and instruction caches as well as the IDC of data and instruction cache and their associated defect map. Figures 14 illustrates the percentage of energy saving of each benchmark when compared to its total energy consumption at the nominal voltage. It is assumed that the IDC and BLB are power gated at nominal voltage in order to eliminate their leakage.

The savings in dynamic energy consumption increase when we reduce the supply voltage but only up to a certain point. We will refer to this point as "turning point". Further lowering the V_{dd} increases the number of defective FGs beyond the tolerance of the proposed architecture and results in an abrupt increase in miss rate, and as a result a sharp increase in dynamic energy consumption. In addition to the voltage level, particular execution properties of each benchmark affect the extent of their energy saving. As stated previously the IDC is much smaller than the total number of defective words in the cache and therefore cannot store all the instructions that are mapped to defective FGs in the cache. Instead, the IDC only keeps those defective words that fall inside the WoE. When the WoE changes, the IDC should be updated with instructions mapped to defective FGs in the new WoE. Based on this discussion one infers that programs whose WoE is rapidly changing (multiple branching, and generally less local), or those that have a very large window (such that the number of defective words inside that window exceeds the IDC size), can not fully benefit from the proposed architecture. Such programs will result in lower energy savings. The "eon" and "crafty" benchmark in Figures 14 are programs with multiple small but long executing loops, small execution window and good locality. On the other extreme, "mcf" and "art" benchmark have fewer localities and rapidly changing execution window. The other benchmarks fall between these two extremes.

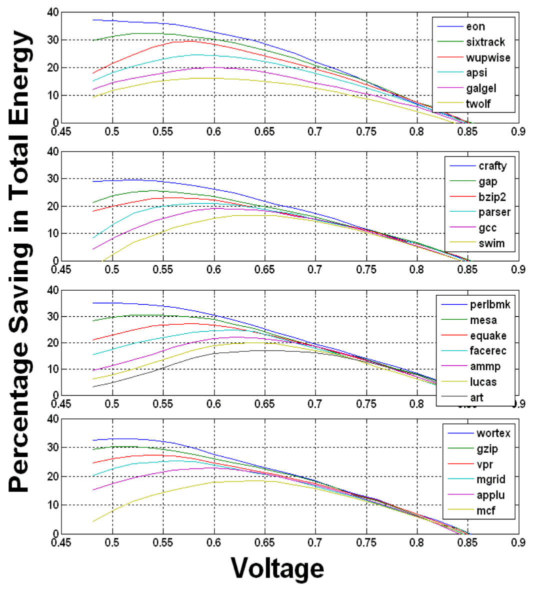

Figure 14: Percentage saving in total energy consumption for different SPEC2000 benchmarks

Note that when migrating to more aggressive nodes (smaller geometry), leakage will become dominant, allowing the IDC "turning point" to occur at lower voltages. However, there will always be a saturation point since the extra soft misses not only increase the dynamic power consumption by requiring access to lower level memories, but also increase the execution time allowing the circuit to leak for a longer period.

The energy savings that are reported in Figure 14 are for a VFS policy defined by parameters (a,b,c) = (1,1,30). By choosing a different set of parameters for (a,b,c) the failure rate and therefore the amount of utilization of IDC at different voltages will change. Table 6 reports the best, worst and average power savings as well as "turning point" voltages associated with different policies (32 nm technology). As a reminder, when a and b are chosen 0, the voltage

*[3] Note that the cell failure curve for L1 and L2 caches could be slightly different due to different bank sizes and implied higher capacitive loads in the bitlines of the L2 cache structure. However, using segmented bitlines, the difference between failure probabilities could be reduced significantly.

scaling policy is FFVS. The last three rows in Table 6 are FFVS associated to failure rates reported in Figure 2, for cycle times of 100, 120, and 160 ps. Note that although going to larger access times in an FFVS policy improves the energy savings, when the access time, become excessively large, this energy saving will not improve as fast and it is even possible to see degradation in total energy saving. This is because although the circuit consumes less dynamic power, however, since the execution time is extended, the circuit will leak for a longer time and therefore the total energy consumption increases. This explains why the energy savings achieved by choosing access time of 160ps is close to that of a 120ps.

In this paper, we have assumed that many techniques such as drowsy cache technique [18] horizontal and vertical sharing of sleep transistor for decoder leakage [19] , etc are in place. If leakage control techniques are not in place, the total energy consumption will increase, resulting in the "turning points" of different benchmarks to be closer to one another. Table 7 provides the same information as of Table 6 for SimpleScalar simulation using 45nm technology failure rates and energy values for cache, IDC and BLB. As reflected in this table the energy savings in 45 nm are smaller compare to that in 32nm. That is because in 32nm we have higher leakage, and since voltage scaling reduces the leakage exponentially it is expected to have higher energy savings in 32nm technology as compared to that in 45nm.

On average, energy savings across all the benchmarks using (a,bc) =(1,1,30) reaches its maximum at 0.58V to be 27.11%. This average maximum energy saving for 32 and 45 nm technology across all SPEC2000 benchmarks when different voltage scaling policies are used is reported in Tables 6 and 7 respectively.

C. Area overhead analysis

Using area figures from CACTI for the main cache (32KB, 4 Banks), IDC (of size 32 rows /2 way associative) and BLB in 32 nm technology, an area overhead of 7.69% is incurred. This area overhead is less than that reported in [3] since the defect map is generated at a larger granularity (FG rather than word) and therefore the area of BLB is reduced. Our layout of an IDC enabled cache (in 45nm technology using TSMC-45LG and ICstudios) resulted in a design with 7.31% area overhead. Note that based on the size of the cache, the technology size that the structure is implemented at, the ratio of the size of IDC to the size of the cache and many other parameters, this area overhead is subject to change. However our purpose in providing the area overhead data is to clarify that the area cost to implement the proposed cache is indeed small and acceptable. Furthermore, it is important to note that the proposed scheme can also be used to correct for manufacturing defects and therefore remove the need for Built-In Self Repair (BISR) (but not BIST), the actual penalty can be much less when compared to memories with BISR.

Table 6: Energy savings for benchmarks with minimum and maximum voltage turning points and average energy saving for all benchmark at average turning point in 32 nm technology

Policy, (a,b,c)	Best%, Voltage	Worse%, Voltage	Average%, Voltage
VFS, (1,1,30)	36.70, 0.48	18.61, 0.67	27.11, 0.58
VFS (1,0,50)	31.14, 0.54	18.21, 0.72	24.64, 0.64
VFS (1,3,0)	33.24, 0.45	25.15, 0.55	29.12, 0.50
FFVS(0,0,100)	31.62,0.55	16.13,0.69	24.11, 0.62
FFVS(0,0,120)	35.56,0.49	29.15,0.58	33.01,0.53
FFVS(0,0,160)	37.16,0.46	26.63,0.63	23.12,0.55

Table7: Energy savings for benchmarks with minimum and maximum voltage turning points and average energy saving for all benchmark at average turning point in 45 nm technology. In 45nm, memory cell access time was measured 120ps with standard deviation of 4.11ps at nominal voltage.

Policy, (a,b,c)	Best%, Voltage	Worse%, Voltage	Average%, Voltage
VFS, (1,1,60)	28.32, 0.44	11.23, 0.64	21.25, 0.54
VFS (1,0,80)	29.78, 0.51	15.44, 0.70	21.36, 0.61
VFS (1,3,0)	31.45, 0.42	21.17, 0.54	25.48, 0.48
FFVS(0,0,200)	26.54,0.52	11.32,0.64	18.45, 0.63
FFVS(0,0,225)	28.32,0.45	16.21,0.54	23.89,0.50
FFVS(0,0,250)	29.03,0.55	18.44,0.60	25.54,0.52

IX. CONCLUSION

In this paper, we targeted caches for aggressive supply voltage scaling. We present a new architecture that uses an auxiliary cache (IDC) and a defect map (BLB) to enable significant overall power savings while maintaining low failure rates. The proposed approach can be extended in many ways, such as considering whole memory hierarchies, and defining policies for voltage-frequency power management that are more effective than traditional ones assuming "defect free" operation. The impact of this technique becomes more pronounced with decreased process geometries.

References

[1] S. Mukhopadhyay, H. Mahmoodi, K. Roy, "Modeling of failure probability and statistical design of SRAM array for yield enhancement in nanoscaled CMOS," Computer-Aided

Design of Integrated Circuits and Systems, IEEE Transactions on , vol.24, no.12, pp. 1859-1880, Dec. 2005

[2] A. K. Djahromi, A. Eltawil, F. J. Kurdahi, R. Kanj, "Cross Layer Error Exploitation for Aggressive Voltage Scaling," Quality Electronic Design, 2007. ISQED '07. 8th International Symposium on , vol., no., pp.192-197, 26-28 March 2007

[3] M. A. Makhzan (A. Sasan), A. Khajeh, A. Eltawil, F. J. Kurdahi, "Limits on voltage scaling for caches utilizing fault tolerant techniques," Computer Design, 2007. ICCD 2007. 25th International Conference on , vol., no., pp.488-495, 7-10 Oct. 2007

[4] http://www.hpl.hp.com/personal/Norman_Jouppi/cacti4.html

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

[6] P. Shirvani, E. J. McCluskey, "PADded cache: a new fault-tolerance technique for cache memories ," VLSI Test Symposium, 1999. Proceedings. 17th IEEE , vol., no., pp.440-445, 1999

[7] H. T. Vergos, D. Nikolos, “Efficient Fault Tolerant Cache Memory Design”, Micro processing and Microprogramming Journal, vol.41, no.2, pp.153-169, 1995.

[8] G. S. Sohi, "Cache memory organization to enhance the yield of high performance VLSI processors," Computers, IEEE Transactions on , vol.38, no.4, pp.484-492, Apr 1989

[9] A. F. Pour, M. D. Hill, "Performance implications of tolerating cache faults," Computers, IEEE Transactions on , vol.42, no.3, pp.257-267, Mar 1993

[10] L. Xiao, J. C. Muzio, "A fault-tolerant multiprocessor cache memory," Memory Technology, Design and Testing, 1994., Records of the IEEE International Workshop on , vol., no., pp.52-57, 8-9 Aug 1994

[11] Y. Ooi, M. Kashimura, H. Takeuchi, E. Kawamura, "Fault-tolerant architecture in a cache memory control LSI," Solid-State Circuits, IEEE Journal of , vol.27, no.4, pp.507-514, Apr 1992

[12] A. Agarwal, B. C. Paul, S. Mukhopadhyay, K. Roy, "Process variation in embedded memories: failure analysis and variation aware architecture," Solid-State Circuits, IEEE Journal of , vol.40, no.9, pp. 1804-1814, Sept. 2005

[13] C. Wilkerson, G. Hongliang, A. R. Alameldeen, Z. Chishti, M. Khellah, L. Shih-Lien, "Trading off Cache Capacity for Reliability to Enable Low Voltage Operation," Computer Architecture, 2008. ISCA '08. 35th International Symposium on , vol., no., pp.203-214, 21-25 June 2008

[14] A. J. Bhavnagarwala, T. Xinghai, J. D. Meindl, "The impact of intrinsic device fluctuations on CMOS SRAM cell stability," Solid-State Circuits, IEEE Journal of , vol.36, no.4, pp.658-665, Apr 2001

[15] http://www.eas.asu.edu/~ptm

[16] A. Agarwal, K. Roy, T. N. Vijaykumar, "Exploring high bandwidth pipelined cache architecture for scaled technology," Design, Automation and Test in Europe Conference and Exhibition, 2003 , vol., no., pp. 778-783, 2003

[17] M. A. Makhzan (A. Sasan), A. Khajeh, A. Eltawil, F. J. Kurdahi, "A Low Power JPEG2000 Encoder With Iterative and Fault Tolerant Error Concealment," Very Large Scale Integration (VLSI) Systems, IEEE Transactions on , vol.17, no.6, pp.827-837, June 2009

[18] K. Flautner, K. Nam Sung, S. Martin, D. Blaauw, T. Mudge, "Drowsy caches: simple techniques for reducing leakage power," Computer Architecture, 2002. Proceedings. 29th Annual International Symposium on , vol., no., pp.148-157, 2002

[19] H. Homayoun, A. Sasan (M.A. Makhzan), A.V. Veidenbaum, “Multiple sleep mode leakage control for cache peripheral circuits in embedded processors” CASES 2008

[20] M. A. Lucente, C. H. Harris, R. M. Muir, "Memory system reliability improvement through associative cache redundancy," Custom Integrated Circuits Conference, 1990., Proceedings of the IEEE 1990 , vol., no., pp.19.6/1-19.6/4, 13-16 May 1990

[21] Z. Bo, D. Blaauw, D. Sylvester, K. Flautner, "The limit of dynamic voltage scaling and insomniac dynamic voltage scaling," Very Large Scale Integration (VLSI) Systems, IEEE

Inquisitive Defect Cache: A Means of Combating Manufacturing Induced Process Variation

Avesta Sasan (Mohammad A Makhzan), Houman Homayoun, Ahmed Eltawil, Fadi Kurdahi,
{mmakhzan,hhomayou, aeltawil,kurdahi}@uci.edu

***Abstract**-This paper proposes a new fault tolerant cache organization capable of dynamically mapping the in-use defective locations in a processor cache to an auxiliary parallel memory, creating a defect-free view of the cache for the processor. While voltage scaling has a super-linear effect on reducing power, it exponentially increases the defect rate in memory. The ability of the proposed cache organization to tolerate a large number of defects makes it a perfect candidate for voltage-scalable architectures, especially in smaller geometries where Manufacturing Induced Process Variation (MIPV) is expected to rapidly increase. The introduced fault tolerant architecture consumes little energy and area overhead, but enables the system to operate correctly and boosts the system performance close to a defect-free system. For a 16KB L1 cache using an Inquisitive Defect Cache (IDC) of size 1KB, power savings of over 40% is reported on standard benchmarks while the performance degradation is maintained below 1%.*

Index Terms – Process variation, Low Power, SRAM, Cache, Fault Tolerance, voltage scaling, power efficient, technology scaling, manufacturing defects.

I. Introduction

The tradition of using minimum sized transistors to create compact SRAM structures, coupled with the analog nature of memory read/write operations, results in a much larger defect density in memories as compared to logic. In addition, it is shown in [1][2][17] and will further be discussed in section III that voltage scaling exponentially increases the memory cell failure probability. This implies that caches, which are the largest memory structures in the processor, abruptly limit the processor lower limit of supply voltage by introducing an exponential relation between their bit failure rate and the system supplied voltage. Since the failure probability under voltage scaling is exponential, fault tolerant techniques, such as row and column redundancy, that linearly increase the fault tolerance limit of the cache will fall short for significant reductions of the supply voltage ($V_{cc}^{\min}$) due to their limited fault coverage. This suggests that a fault tolerant mechanism, to be used in voltage scaled caches, should be able to tolerate a very large number of defects. The increase in fault coverage however should be achieved in a fashion such that 1) the impact of fault handling mechanism on delay and/or performance of system is acceptable, 2) the total power consumption of the system including the new fault handling mechanism is lower than that of other simpler fault handling mechanism that operate at higher voltages handling lower number of defects with lower power/overhead.

In this paper, we explore the organization of a novel fault tolerant cache architecture which is capable of providing fault coverage for caches suffering from very high defect rate. We first introduced this organization in [3] for direct mapped caches with single word fetch per cache access. In this work, we further explore the usage of the (Inquisitive Defect Cache) IDC to associative caches with multiword fetch groups, which resulted in significantly smaller area overhead. Furthermore, we introduce a mathematical upper bound on defect coverage of the proposed architecture relating the area and power of the proposed architecture to achievable V_{cc}^{min}. In [3] we investigated the fault coverage and voltage scalability of the system when a Fixed Frequency Voltage Scaling (FFVS) policy for scaling the cache supplied voltage is used. In this work, we extend our study and results examining the utilization of the proposed architecture for both Voltage Frequency Scaling (VFS) and FFVS policies. In [3] for obtaining the power and leakage figures we utilized CACTI [4] power and leakage estimations, however in this work, in order to improve the accuracy of our results, the leakage and power figures are replaced with those obtained from post layout simulation of the cache layout. Furthermore, the simulation environment (DINERO IV) presented in [3] has been replaced with SimpleScalar [5] to allow simulation of the architecture for well known SPEC2000 bench marks.

The paper is organized as follows: Section II builds the necessary background and highlights the motivation behind this work. Section III explains the behavior of memory cell under voltage scaling exploring the effect of process variation in lower voltages and tradeoff between probability of cell failure rate, voltage scaling and frequency scaling. Section IV explains the effect of voltage scaling on memory logic. Section V introduces our proposed architecture which guarantees the correct operation of the cache and at the same time creates a virtually defect-free view of the cache for the processor, Section VI introduces an energy model to quantify the energy savings in lower voltages. Section VII presents a mathematical bound on IDC coverage. Section VIII briefly refers to future directions in our research, where we investigate usage of the IDC to tradeoff speed versus power. In section IX, we validate our proposed concept trough SimpleScalar simulation, illustrating how it could reduce overall system energy and power consumption by tolerating a larger numbers of defects and therefore allowing better performance at nominal voltages or alternatively the same performance at lower voltages. Finally, the paper is concluded in section X.

II. Prior Work

Cache is a high speed SRAM memory used to bridge the speed gap between the processor and slower Lower Level Memories (LLM). For accessing LLM, one pays the price of longer access time and large power consumption. By reducing the number of accesses to LLM, cache decreases the access time and improves the overall dynamic power consumption. It is the temporal and spatial locality of instruction and data that allows the architect to use caches for holding data and instructions closer to the CPU. If a cache is defective, the system could still operate correctly provided that the faulty cache block or word could be disabled. One way to achieve this is by marking a defective cache word/block with an extra bit that can be added to the set of flag bits of that block, conditioned that the added bit is not defective. In [6] this bit was referred to as the Fault Tolerance bit (*FT-bit)*. The set of FT-bits for memory words or blocks is called the *defect map*. This defect map is used to turn a cache line off in case it is faulty. Turning a cache line off in an associative cache reduces the degree of associatively by one. In a direct mapped cache, on the other hand, every access to a disabled line will result in a miss. This scheme of disabling faulty blocks is done even when the cache is protected by *single-error correcting, double-error detecting* (SEC-DED) codes [7].

Replacement techniques, such as extra rows and columns, are also used in caches for yield enhancement [20] and for tolerating lifetime failures [8][9][10][11] With replacement techniques, there is no performance loss in caches with faults. However, the number of extra resources limits the number of faults that can be tolerated using these techniques. Another form of redundancy is the use of extra bits per word to store an error correcting code. Sohi[8] investigated the application of a Single Error Correcting and Double Error Detecting (SEC-DED) Hamming code in an on-chip cache memory and found out that it degrades the overall memory access time significantly.

The work in [12] suggested resizable caches. In this technique it is assumed that in a cache layout, two or more blocks are laid in one row, therefore the column decoders are altered to choose another block in the same row if the original block is defective. In a technique called PADded caches [6] the decoder is modified to allow remapping of the defective blocks to other locations of the cache without destroying the temporal locality of data or instruction. A recent paper from Intel’s microprocessor technology lab [13] suggested the use of fault tolerant mechanisms trading off the cache capacity and associatively for fault tolerance. The proposed approaches while reducing the frequency allows scaling the voltage from a nominal 900mv down to 500mv in a 65nm technology. The cache size is reduced to 75% or 50% depending on the mechanism that is used. In this scheme, data that is mapped to defective location is reallocated to healthy locations. The delay overhead, and the smaller cache size suggests that this scheme could not be used when high performance is desired, since, not only do we have to reduce the frequency, but also, due to smaller cache size, the number of cache misses will increase.

In this paper, a new architecture is presented specifically addressing power and yield improvement via fault tolerance. The ability to tolerate process and operational condition induced faults allows aggressive voltage reduction even in high performance mode (Fixed Frequency), which in turns leads to reduced power consumption

III. Effect of Voltage Scaling on Memory

A. Classification of Memory Errors

Memory cell failures are generally divided into three distinct categories: (1) Fixed Errors or Manufacturing defects, which are unaccounted shorts and opens in the memory cell and its supporting logic resulting in erroneous operation. (2) Transient Errors which are temporary memory malfunctions as a result of a transient situation such as an alpha particle attack. (3) Operating condition dependent due to manufacturing process variation in device physical parameters. This group of failures is not always present however with change in the memory operation conditions including: Process, Voltage & Temperature (PVT) their number varies. Symptoms of these failures are change in cell access times, or unstable read/write operations to some locations upon change in PVT.

Significant contributors to the rapid increase in the number of process variation induced defects include line roughness, oxide thickness variations, as well as Random Dopant Fluctuations (RDF) which results in mismatch between adjacent transistors threshold voltage (Vth) [1]. We will refer to such variation as Manufacturing Induced Local Process Variation (MILPV). When applied to memory cells, the Vth variation results in large variation in access and write time to the memory cells. The dependence of Vth on temperature makes the write/access time also sensitive to die temperature. Furthermore, since the transistor speed is a strong non-linear function of Vdd-Vth ,the access/read time is also strongly and non-linearly dependent on the supply voltage.

B. Modeling Memory Operations Time under Process Variation

In this study 3 types of memory failures were investigated including Read Failure (RF), Write Failure (WF) and Access Failure (AF):

1. ***Access failure*** occurs when in a given access time (from activation of wordline to activation of sense amplifier) the cell can not create enough differential voltage between bitlines.
2. ***Write Failure*** occurs when in a given write time the cell can not be written.
3. ***Read Failure*** is defined as a read operation that destroys the content of the cell.

We setup an experiment to study memory failures due to process variability. Adopting from [22][14] due to RDF effect, threshold voltage variation is considered as the primary source of device mismatch. The circuit under test is a standard six transistor SRAM memory bit cell in a sub-bank of 64 cells connected to the same bitlines. The SPICE models used for the simulation were obtained from the Predictive Technology Model (PTM) [15] website in 32nm. The σ_{Vth} is expected to be ~ 34mV [23] in 32nm technology. Monte Carlo simulations were performed with 10^7 iterations at each voltage level with a different threshold voltage assigned to each transistor in the SRAM cell based on a Guassian distribution. At each simulation a simple Marching algorithm is performed as follows:

- ***Write logic '1' to the cell (initialization)***
- ***Write logic '0' to the cell***
  - *Measure time to write logic '0',*
  - *Finish SIM either on successful "write" or t = T_max_write*
- ***Read a 0 from the cell***
  - *Measure time to create differential voltage Vdiff*
  - *Measure time to bit flip if happened*
  - *Finish SIM either on "bitflip" or t = T_max_read*
- ***Write a 1 to the cell***
  - *Measure time to write logic '1',*
  - *Finish SIM either on successful "write" or t = T_max_write*
- ***Read a 1 from the cell***
  - *Measure time to create differential voltage Vdiff*
  - *Measure time to bit flip if happened*
  - *Finish SIM either on "bitflip" or t = T_max_read*

As a first result from this experiment, access and write at each voltage point are extracted. We observed that the extracted data could be estimated by a Gaussian curve in the range of -2σ to +2σ, however outside this range data has smaller decay compared to a Gaussian curve. Table 1 illustrates the mean and standard deviation of the fitted curves across different voltages for write and read time.

Table 1: Change in the mean and standard deviation

Voltage	Access Time		write Time	
	μ	σ	μ	σ
0.9	38.21	6.92	28.46	5.51
0.85	49.5	9.93	34.34	7.46
0.8	58.63	12.65	42.65	9.45
0.75	70.26	16.74	50.1	12.81
0.7	83.24	19.92	61.03	15.58
0.656	102.78	25.65	73.18	18.36
0.6	125.35	31.34	91.04	21.35
0.55	145.11	36.45	110.25	25.41
0.5	183.43	46.98	131.77	29.26
0.45	233.59	64.68	150.16	35.82

From this table, we observe that when going to lower voltages not only do the mean access and write time increase, but their standard deviations also widen. The next step is to measure the probability of failure. Please note that, the results presents identify the expected trends. Results will vary by changing the sizing of the cell, architecture of the memory bank, strength of the write transistors or by using a different model card. The trend of change in the mean and standard deviation, and association of the failure rate to voltage and access time (as discussed in the next section) will not change.

C. Defining System Cycle Time, Safety Margin (SM) and its Implications on Memory Cell Failure Rate

The failure probability curves are derived based on a voltage frequency policy. A voltage frequency policy determines the T_{Access} and T_{Write} at each voltage point. For a non pipelined cache T_{Access} and T_{Write} are the total access time and total write time where as for a pipelined cache T_{Access} terms represent the time for the memory stage of the cache access from wordline activation to sense amplifier activation. And T_{Write} represent the time from wordline activation to completion of write. Same argument holds for mean access time μ_T^{Acc} and mean write time μ_T^{Wri} .

At each voltage T_{Access} and T_{Write} should be accordingly larger than mean access time μ_T^{Acc} and write time μ_T^{Wri} at that voltage meaning:

$$T_{cycle} = \mu_T^{Acc} + T_{SM}^{Acc} = \mu_T^{Wri} + T_{SM}^{Wri} \quad (1)$$

Where, T_{SM}^{Acc} & T_{SM}^{Wri} are the Access and Write safety margins.

By assigning different values to T_{cycle} we can define different VFS policies. Choosing a large T_{cycle} result in a large T_{SM}^{Acc} & T_{SM}^{Wri} and therefore larger protection against process variation but also degraded performance. On the other hand, a small cycle time reduces the safety margin and increases the probability of failure. Knowing the cycle time at each voltage point from the Monte Carlo simulation results of section C for each (voltage, frequency) a probability of failure could be extracted. The following equation allows us to define different VFS policies.

$$T_{cycle} = a \times Max(\mu_T^{Acc}, \mu_T^{Wri}) + b \times Max(\sigma_T^{Acc}, \sigma_T^{Wri}) + c \quad (2)$$

In this equation a, b, and c are scale factors and σ_T^{Acc} & σ_T^{Wri} are the standard deviation of access time and write time accordingly. At each voltage the associated mean and standard deviations at that voltage level are used.

If coefficients "a" and "b" are set to zero, a special case of VFS is created, which we refer to *Fixed Frequency Voltage Scaling (FFVS)*. In this policy when lowering the voltage, the cycle time stays constant. In this manner the increase in the mean access and write time reduces the SM and therefore reduces the safety margin. This results in a rapidly increasing defect rate, albeit, maintaining the memory performance. The advantage of this type of voltage frequency scaling is that the cycle time is constant and if some Fault Tolerant mechanism could handle or mask the increase in the failure rate, a high performance system at lower power is obtained. Figure 1 depicts the probability of failure for such a scenario.

Equation (2) creates an expanded design space where different policies can be envisioned based on different values of (a,b,c) as shown in Table 2. Using data from Table 2 the probability of failure at different voltages associated with each VFS policy was extracted and representative schemes depicted in Figure 2. The probability of failure is the accumulated sum of all three types of failures (FR, RF, AF).

As illustrated, VFS could be performed in numerous ways. The defined frequency of operation at each voltage point relates the probability of failure to that specific voltage level. By choosing a larger access time (and therefore larger safety margin) at lower voltages, the system's ability to tolerate process variation is increased; however the increased system cycle time deteriorates performance. The implication on power consumption is twofold. On one hand, the lower voltage results in lower dynamic and leakage power consumption. On the other hand, the lower performance and the resulting larger execution time imply that the circuits run for a longer time, which increases the total energy consumption. The appropriate lower limit on the voltage should be chosen by considering the maximum tolerable defect rate, the desired performance and power consumption.

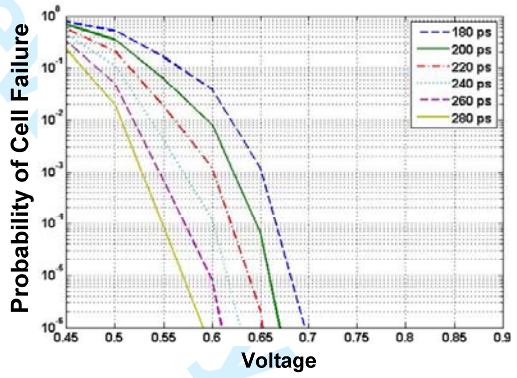

Figure 1: Probability of cell failure for FFVS for different choices of c in equation 2

Table 2: Cycle time for different choices of (a,b,c)

	1.5	**1.5**	**1**	**1**	**0**	**1**	**0.5**	**a**
	0	**1.5**	**1**	**2**	**2**	**0.5**	**2**	**b**
Voltage	**60**	**0**	**70**	**80**	**100**	**100**	**50**	**c**
0.9	117	67	115	132	114	141	83	
0.85	134	88	129	149	120	154	94	
0.8	147	106	141	163	125	164	104	
0.75	165	130	157	183	133	178	118	
0.7	185	154	173	203	140	193	131	
0.656	213	191	198	233	151	215	152	
0.6	248	235	226	268	163	241	175	
0.55	278	272	251	297	172	263	195	
0.5	335	345	300	357	194	307	236	
0.45	410	446	367	441	228	365	295	

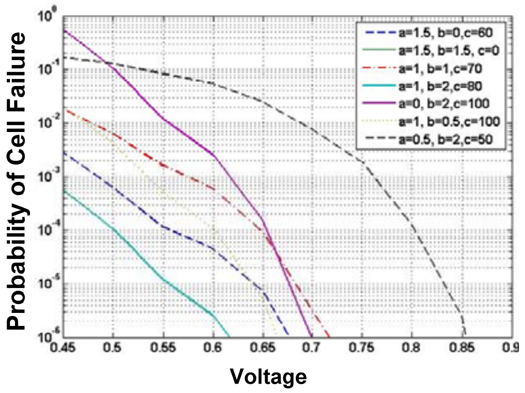

Figure 2: Probability of cell failure for VFS for different choices of a, b and c in equation 2

IV. Effect of Voltage Scaling on Memory Logic

In this section, we focus on a three stage pipeline cache as a case study, but note that the same arguments could be applied to wave pipelined and non-pipelined caches as well. We specifically choose the 3 stage pipeline cache to show some of the design considerations when a system is designed for FFVS or VFS policy*[1].

In a three stage pipeline cache, the cache operation is divided into; a) Decoding in the first stage, b) Memory cell access in the second stage, which is considered from activation of the wordline, to the analog operation of reading the target memory cells into bitlines and ends with sense amplifier activation and, finally c) Sensing, muxing and driving the output signals in the third stage. In this configuration, the second stage (reading the memory cells) is the longest pipeline stage. This stage is analog and cannot be pipelined into further stages [1]. Figure 3 illustrates the timing of a three stage pipelined cache operating at nominal V_{dd}. When pipelined, every stage in the cache has one cycle to complete its operation. The cycle length is determined by the delay of the longest stage in pipeline.

In an FFVS system the second stage delay stays constant throughout the voltage scaling and the shift in the distribution of the access/write time is translated to higher failure rates. One determinant for the lowest possible voltage $V_{cc}^{\min}$ is the tolerable failure rate. Another consideration for such system is the design of other stages of the pipeline (first and third stage in this case). These stages are purely logic stages. Reducing the voltage increases the logic propagation time and therefore reduces the operational speed of these stages. The delay of these stages, however, cannot exceed the delay of the second stage at lower voltages. Therefore, FFVS system must be designed such that all logic operations complete correctly at the lowest voltage. This means potentially using larger and faster transistors in the design.

On the other hand, if a VFS policy is used; as the voltage is lowered, the second stage delay (Δ) increases. Since all the stages of the pipeline should take equal time, the delay of the second stage is multiplied by the number of stages in the pipeline, increasing the access time by $N.\Delta$ and the cycle time by Δ. In such a case, the delay slack in the first and third pipeline stages allows the usage of smaller transistor in those stages to achieve lower power consumption.

The delay of the logic [21] in the first and third stages is related to the supply voltage by the following equation:

$$Delay = \frac{V_{dd}}{(V_{dd} - V_{th})^{1.3}} \qquad (3)$$

Table 3 depicts the delay of various stages in a 16KB cache. The delays are obtained by simulating a post-layout realization of a 16KB cache whose parameters are described in the same table. Date presented in table 3 includes the delay of the decoder, memory cell, and the output logic for both tag and data.

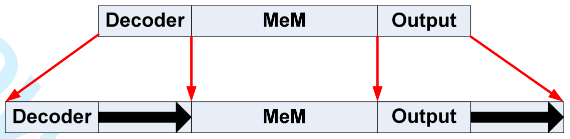

Figure 3: Converting the access time of a non pipelined cache to a pipelined cache

In this paper, we assume that all memory components are controlled by a single variable supply voltage. This assumption is made for the sake of simplicity and implementation purposes. The first and third stages have a lower limit on how much their voltage could be reduced before the stage delay exceeds the second stage delay (which defines the cache cycle time). In the proposed cache, when reducing the voltage, the tag decoding stage delay reaches the cycle time limit before other logic stages and since all stages are connected to a single supply voltage, this voltage defines the lower limit of supply scaling $(V_{cc}^{\min})$.

Table 3: Nominal timing of a pipeline cache of size 16KB with 1024 sets, associativity of 4 and block size of 16 bytes in 32nm.

Data Delay (ps)		Tag Delay (ps)		Stage	Delay(ps)
decode	13	decode	17	1	17
wordline	16	wordline	14	2	38
Bit-line	22	Bit-line	21		
sense amp	7	Sense amp	6	3	18
Output	15	Output	18		

*[1] Although FFVS could be considered as a special case of VFS however the design implications-- especially in pipeline structures—warrant considering FFVS separately.

V. Proposed Architecture

Figure 4 illustrates our purposed architecture. A cache in this model is augmented with two auxiliary blocks. The Bit Lock Block (BLB) is a defect map for the cache and the IDC is a small auxiliary cache for the purpose of defect tolerance. Both of these auxiliary structures are kept at high supply voltage when the voltage of the main cache is scaled. This is to guaranty reliable storage of defect information in the defect map. The size of these two structures compared to cache is much smaller and it will be shown that their augmented power penalty is much smaller than that saved by allowing aggressive voltage scaling within the main cache.

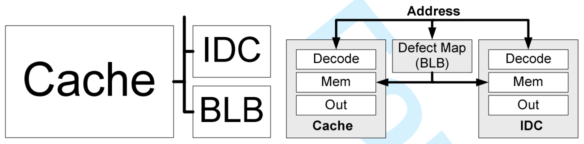

Figure 4: Proposed architecture

A. Bit Lock Block (BLB) defect map

The number of words that are read from a cache at each read cycle is referred to as a Fetch Group (FG). In this architecture, to create a defect map, we use one bit for each FG in cache, which is stored in the BLB. The size of the BLB is therefore proportional to the size of the cache and is fixed. Unlike some previous implementations of defect map for caches, where a portion of the tag bits were used for keeping the defect map, we separated the defect structure from the tag section to allow scaling the voltage of the tag bits along with memory bits while the defect information is kept at a high supply voltage. This guarantees a smaller structure for the defect map and faster access to the defect information.

B. Generating and Updating the BLB

As part of the operation of the system, we need to have a mechanism for generating the BLB and another one for updating it. Generating the BLB could be accomplished at manufacturing time, however testing for multiple vectors significantly increases the chip cost. Another approach is to generate the defect map at boot time. In this method, at manufacturing time, the memory is only tested for the manufacturing defects at the nominal voltage process corners, fixing such errors using the available redundancy and leaving the process variation induced defects to the combination of the (Fault Tolerant Memory) FTM and Built In Self Test (BIST) unit. At run time BIST tests the memory arrays at lower voltages and write the defect map into BLB. BIST might generate a different BLB for operation at different PVT corners. Defect maps can be saved by the system in non volatile memory to avoid running the BIST in the future or can be re-run on demand.

To maintain current information in the BLB that takes into consideration operation environment changes (e.g. temperate), we need to consider a capability for dynamically updating the BLB. This is achieved in one or a combination of the following ways: (1) Usage of parity bits: Every time that a new error is discovered, its location is registered in a very small memory (in our case 16 or 32 entry). Since the newly discovered defect could be a soft error, BLB is not updated immediately. If the defective FG is detected again then it is considered a temperature induced error and is registered in BLB. Each BLB row has a dirty bit which is set if a bit in that row is updated. The presence of the dirty bit requires a write back of the BLB information when changing the voltage level. (2) Run the built in BIST engine periodically. (3) Using Dynamic Temperature Sensing (DTS) infrastructure, an approach which is already being adopted by many manufacturing companies in 65nm and technologies and below. DTS is geared towards avoiding overheating but can also be used for other purposes such as modifying the policy stacks. In this approach, the system senses the cache temperature. When a temperature increase exceeding a pre-specified amount is detected, the BIST engine is instructed to test the heated locations for temperature induced defects. The BLB is updated and the dirty bit is set. Since temperature is a very slow changing variable such BIST updates will not run frequently. In addition, as the chip ages the BLB coverage increases and less BLB updates are required.

Every time the voltage changes, the incremental defect maps are loaded into BLB. If there are any dirty bits a write back of BLB information is needed before loading the new defect map. The BLB itself is very small compared to the cache and therefore it can operate much faster than the cache. Furthermore, the BLB voltage is not scaled, therefore its delay never exceeds the cache cycle time.

Although IDC is designed for dealing with process variation defects, it could also be used to deal with manufacturing defects in upper level memories (caches). In other words, manufacturing defects could be considered as process variation defects that are present at all corners of the PVT space. With this in mind if at the manufacturing the number of available redundancy is less than the number of manufacturing defects, conditioned that their number is smaller than available space in IDC and they are spatially distributed in FGs that are not in close proximity, they could be masked by the proposed mechanism.

C. *Inquisitive Defect Cache memory*

The Inquisitive Defect Cache (IDC) is a small associative cache which temporarily holds FGs mapped to defective locations in its associated cache while these FGs are within the Window of Execution (WoE). The conceptual view of an IDC enabled cache structure is illustrated in Figure 5.

The size of IDC cache could be much smaller than the total number of defective FGs in a cache. In a sense, the IDC could be considered as a group of redundant rows with the main difference being that mapping to redundancy is fixed and one time, whereas mapping to the IDC is dynamic and changes with changes in the addressing behavior of the program. Furthermore, in the IDC case, the defective FGs within the Window of the Execution (WoE) of the program in the cache are mapped to IDC allowing much higher coverage than that of a redundancy scheme.

In Figure 5, IDC, Cache and BLB represent the Inquisitive Defect Cache, Cache and Defect Map (Bit Lock Block) respectively. The WoE and "Int" are conceptual structures (do not exist but drawn to ease the understanding of IDC structure and its operation) representing the Window of Execution of the program in the cache and Intersection of the WoE and BLB respectively. The BLB marks the defective FGs in the cache. The WoE marks the in-use section of the cache (FGs that the processor has recently visited). The "Int" which is the intersection of BLB and WoE is then the defective FGs in the WoE which should be mapped to IDC. In Figure 5 the IDC and cache have associativity of 2 and 4, respectively.

If the IDC size and associatively are chosen properly, the WoE of a process in the cache (after its first pass) should experience very few defective/disabled words and could be virtually viewed as a defect free segment in the cache by the processor.

If a FG is defective, its information could only be found in the IDC since every read/write access to defective words is mapped to the IDC. The BLB should be designed such that its access time is smaller than the cache cycle time. Considering the small size of the Defect Map when one bit per FG is chosen, and also the fact that the Defect Map voltage will not be scaled, makes it fairly easy to meet this design constraint.

On each access, all three components (BLB, IDC and Cache) are accessed simultaneously. However immediately after BLB feedback the operation on either cache or IDC for the next cycle is gated to save power. The following section analyzes the access scenarios and derives models for energy consumption for each scenario.

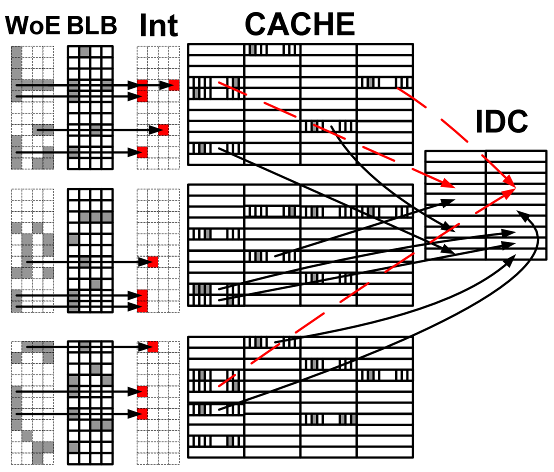

Figure 5: In the first pass defective FGs in window of execution are mapped to IDC.

VI. Energy Models

A. *Cache access scenarios*

An access to the cache, as illustrated in Figure 6, initiates parallel access to cache, IDC & BLB. In this model however after the first cycle, based on defect map feedback, access to either cache or IDC is gated. The dynamic energy consumption when IDC or cache is gated is presented in equations (4) and (5).

If for some design reason the defect map access, could not fit in one cycle, a buffer could be added to act as a cache for the BLB. The buffer consists of a set of flip flops and each access to the BLB is serialized with access to the BLB buffer. In this case, the access could either be gated at the end of Cycle1 (if information is in the buffer) or Cycle2 (if the information in not in buffer and BLB should be accessed), while stopping further access to either the cache or IDC [3]. The timing of this sequence of operations is given in Figure 7. Due to temporal and spatial locality of access, the defect map query is usually resolved by the buffer, saving time and energy even with a BLB that takes longer than one cycle to access. However, in this paper, we build our system based on a BLB capable of providing the defect information in one cycle. The reader is referred to [3] for further detsils on scenarios considering the BLB buffer.

$$E = (E_{BLB} + E_{Didc}) + (E_{Rc}) * \frac{V_{dd}^{\ 2}}{V_{dd0}^{\ 2}} \tag{4}$$

$$E = (E_{BLB} + E_{IDC}) + (E_{Dcache}) * \frac{V_{dd}^{\ 2}}{V_{dd0}^{\ 2}} \tag{5}$$

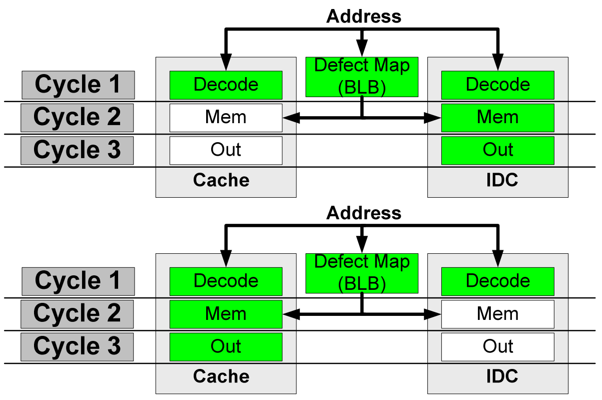

Figure 6: Cache access scenarios (top): defect map refer to IDC, (bottom): defect map refer to cache.

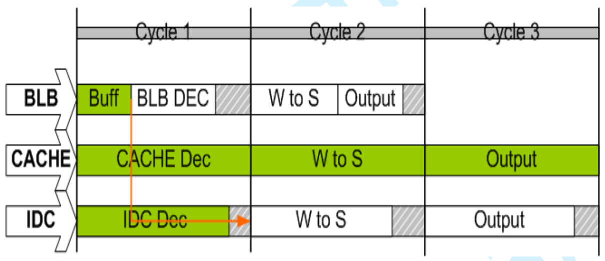

Figure 7: cache access scenario with a Buffer

In Section VIII, we report the simulation results of the proposed architecture in 32 nm technology. We use SPEC2000 bench marks to evaluate the system performance under different input conditions. We evaluated the tolerance of IDC at different voltage points under different voltage scaling policies. For each simulation point, a probability of failure per memory cell is obtained. Using that probability of failure we have created 250 defective caches and simulated the behavior of the proposed architecture. In order to quantify the energy savings, we considered both dynamic and leakage energy consumption. In the following, we illustrate how dynamic and leakage power consumption are obtained.

B. *Dynamic cache energy model*

As a program shifts its WoE to a new location in cache, it may access defective locations that are not currently in the IDC. At this point, the cache will declare a miss and data is obtained from lower levels of memory hierarchy and placed in IDC. For future reference, we define such a miss (a miss due to access to a defective location in the cache which is not covered in IDC) as a "*soft miss*". A soft miss could also occur if the number of defective FGs that are mapped to one physical row in the IDC exceed the associatively of the IDC (this case is illustrated using dashed lines in Figure 5. A soft miss is handled similar to a miss by accessing and retrieving the information from a lower level memory. However in case of a soft miss, the retrieved data should be written to the IDC and not to the main cache since the target location in main cache is defective.

Dynamic power is affected by the miss rate. As the miss rate increases, the dynamic power consumption increases because of the increased access to next level caches and main memory. At the same time a high miss rate could cause the processor to stall, increasing the execution time. This in turn means that the processor and memory would run for a longer time, increasing the total energy consumption. Considering that in smaller geometries leakage energy is comparable and even predicted to be dominant compared to dynamic energy, a considerable energy overhead is expected.

Consider a cache access scenario that results in a miss: The address is sent to BLB, IDC and Cache. All three components start decoding the address at the same time. Following one of the two cases explained in section VI, a miss will occur, at which point, the lower level memory is accessed for data. When data is ready in the lower level cache, it is transferred to upper level either by writing to IDC and/or Cache. When data is updated in the upper level memory an interrupt is sent to the processor and the cache request is reinitiated. At this time the BLB, IDC and Cache are accessed again in parallel. Similar to the initial cache access, one of the 2 access scenarios explained in section VI will now find the data either in the cache or the IDC. To quantify the energy expenditure, all terms used are detailed in Table 4, where the total energy per miss can be calculated as follows:

$$E_{miss} = E_{blb} + AorB(E_{Rmc} \times \frac{V_{dd}^{2}}{V_{dd0}^{2}} + E_{Didc}, E_{Rmidc}$$
$$+ E_{Dc} \times \frac{V_{dd}^{2}}{V_{dd0}^{2}}) + E_{mem} + E_{blb} +$$
$$AorB(E_{Wc} \times \frac{V_{dd}^{2}}{V_{dd0}^{2}}, E_{Widc}) +$$
$$AorB(E_{Rc} + \times \frac{V_{dd}^{2}}{V_{dd0}^{2}} E_{Didc}, E_{Ridc} + E_{Dc} \times \frac{V_{dd}^{2}}{V_{dd0}^{2}}) \qquad (6)$$

The $AorB$ function dictates that only one of its fields will be involved in the energy equation.

$$AorB(E_{Wc} \times \frac{V_{dd}^{2}}{V_{dd0}^{2}}, E_{Widc}) \qquad (7)$$

For example, AorB in Equation 7 indicates that the energy that has occurred during the write is either the write energy to the IDC or scaled write energy to the cache (scaled to factor the voltage scaling effect).

Table4: Terms and variables used

E_{blb} =	Energy per BLB read.
E_{mem} =	Energy of access to the lower level memories
E_{Rc} =	Energy consumed to read cache
E_{Wc} =	Energy of a write to cache
E_{Ridc} =	Energy to read IDC
E_{Widc} =	Energy to write to IDC
E_{Rmc} =	Energy to read cache when miss (no output driver activation)
E_{Rmidc} =	Energy to read IDC when miss (no output driver activation)
E_{Dc} =	Energy to decode cache
E_{Didc} =	Energy to decode IDC
$AorB(A,B)$ =	A If case A happens, B if case B happens
V_{dd} =	Scaled memory supply voltage
V_{dd0} =	Nominal memory supply voltage
L =	Size of the Window of Execution
A =	Associability in the WoE
S =	Associability of the IDC cache
D =	Size of the IDC cache
P_f	Probability of Cell failure
W =	Number of Words in each fetch group
T =	Number of bits in each word
P_{clean_access} =	Probability of clean access

C. *Leakage power analysis*

As discussed previously, leakage energy consumption is also affected by the miss rate. A miss could potentially increase the program execution time and therefore the circuit would leak for a longer period of time. Memory leakage power consumption is exponentially related to the memory supply voltage. At process technologies of 65nm and below, the dominant leakage components are sub-threshold leakage and gate leakage. The gate leakage with usage of high k material will be reduced by a factor of ~100x Therefore we only consider the threshold voltage. The sub-threshold leakage in MOSFET transistors relates to voltage by:

$$I_{leakage} = \mu_0.C_{ox}.\frac{W}{L}.e^{b(V_{dd}-V_{dd0})}.v_t^2.(1-e^{\frac{-V_{ds}}{v_t}}).e^{\frac{-\Delta V_{th}}{n.v_t}} \quad (8)$$

In this equation, $v_t = KT/q$ is the thermal voltage, W is the device width, V_{ds} is the drain to source voltage, n is the drain induced barrier lowering (DIBL) coefficient, and $\mu_0.C_{ox}$ are constants that do not depend on temperature or voltage. The constant term b in the equation is a technology dependent coefficient which is obtained empirically. CACTI4.2 [4] assumes b = 1.7 in 65 nm technology. For 32nm, we ran SPICE simulations of a 6T memory cell using *W/L* ratio of 2, and choosing the widths of Pre-charge and Write NMOS transistors as 16X the size of memory cell NMOS width. Using linear regression on the simulation results, we obtained an estimate of 2.1 for b in 45 nm technology and 3.4 for 32nm technology. In this equation, everything is constant except W, V_{dd}, T and V_{th}. Therefore the equation could be written as:

$$I_{leakage} = W.I_l(T,V_{dd},V_{th}) \quad (9)$$

Using PTM V1.0 [15] and considering the cache to be operating at 360º K, the total leakage of our proposed cache architecture at different voltage levels for 32 and 45 nm technologies is compared to the leakage of traditional cache architecture and is illustrated in Figure 8. While reducing the voltage results in an exponential reduction in the sub-threshold leakage current principally due to the Drain Induced Barrier Lowering (DIBL) effect, it also results in a substantial reduction in gate leakage due to a reduction in the number of tunneling electrons [24].

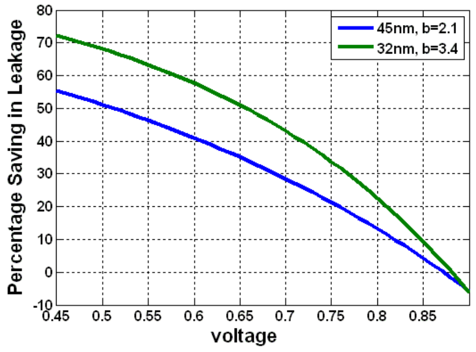

Figure 8: Percentage saving in the leakage power in proposed architecture compared with the traditional cache architecture

At nominal voltage the leakage saving compared to a traditional cache (with no IDC and BLB) is negative because of the additional leakage overhead from IDC cache and BLB*[2]. However, at lower voltages, (as will be shown in the results section (section VIII)) saving in the leakage of cache far outweighs the extra leakage expensed. Notice that only the voltage in the main cache is scaled while the BLB and IDC are kept at their nominal voltages. The leakage savings in the 32nm grows more rapidly than the 45nm technology since the b factor in Equation (8) in 32nm technology is

*[2] It is also possible to completely shut off those components when operating at nominal Vdd therefore eliminating their leakage using power supply gating transistors.

higher than in 45nm technology. (Both technologies are simulated using identical threshold voltages).

VII. MATHEMATICAL BOUND ON IDC COVERAGE.

In this section, we develop the upper and lower bounds on the fault tolerance capabilities of the IDC. In section VIII, it will be illustrated that some benchmarks utilize the IDC better than others. The amount of locality, the size of the WoE, and the sequentiality of the program in execution all contribute to better or worse utilization of the IDC. Considering uniform distribution of MILPV defects in the cache, programs with smaller windows of execution, could better utilize the IDC cache structure. The smaller the WoE, the smaller the expected number of defects in the WoE. This improves the chances that the faults in that window are covered. In addition, the higher the locality of the information, the higher the chances that a previously mapped fault to IDC is used before it is overwritten. Finally, for sequential code (codes where the addressing sequence of their program binary within the WoE are sequential (as opposed to hyper branched)) there is uniform chance of mapping conflict across all rows in IDC improving the overall utilization of IDC. This is because as illustrated in Figure 9, only a subset of locations in the cache could be mapped to each location in IDC.

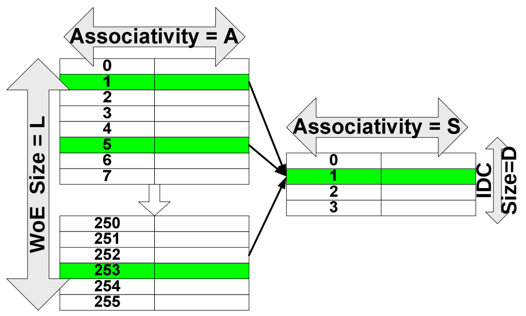

Figure 9: sequential mapping of the cache to the IDC

In order to demonstrate two extremes of fault tolerance performance, we consider two program cases: one with hyper branched addressing (almost every instruction is a branch) and the other with sequential program binary addressing. The probability of a fetch group failure (a defective FG) is:

$$P_{FG} = 1-(1-P_f)^{W.T} \tag{10}$$

We define a clean access as "an access to one line in the WoE which is either mapped to a defect-free FG in the cache or to a FG previously mapped to the IDC". Note that such access could still result in a miss however the miss is not due to addressing a defective FG, but rather to a failure in tag matching (i.e. the intended information is not in the cache).

Based on Figure 9, if the binary in the window of execution is highly sequential, $L.A/D$ lines in WoE (uniform distribution) could be mapped to each line. In this case, access to one line is clean if either the FG is clean, or if it is defective and is already mapped to the defective location and is not in a mapping conflict where more than 'S' FGs that need to be mapped to one line in the IDC. In another words, there is less or equal to *S-1* defects in the other $L.A/D$ lines that are mapped to the same physical address in the IDC that the current FG need to be mapped to. Thus, we have:

$$P_{clean\_access} = P(defect\_free\_FG) + (1-P(defect\_free\_FG)).P_{\text{cov}} \tag{11}$$

In which P_{cov} is the probability that there are less that S other lines mapped to the associated line in IDC.

$$P_{\text{cov}} = P_{MFG}(0)+P_{MFG}(1)+\ldots P_{MFG}(S-1) \tag{12}$$

In which $P_{MFG}(i)$ stands for the probability that '*i*' other FGs in the WoE are mapped to that exact same location in IDC. Therefore:

$$P_{\text{cov}} = (1-P_{FG})^{\frac{L.A}{D}-1} + \binom{\frac{L.A}{D}}{1}.P_{FG}.(1-P_{FG})^{\frac{L.A}{D}-2} + \ldots + \binom{\frac{L.A}{D}}{S-1}.P_{FG}^{S-1}.(1-P_{FG})^{\frac{L.A}{D}-S} = \sum_{i=0}^{S-1}\binom{\frac{L.A}{D}}{i}.P_{FG}^{i}.(1-P_{FG})^{\frac{L.A}{D}-i-1} \tag{13}$$

For the case of hyper branched binary addressing of the program within the WoE, each of the lines in the window of execution is equally likely to be mapped to any line in the IDC. Therefore the probability of a clean access to one line is obtained by replacing $L.A/D$ in equation (13) by $L.A$.

The probability that every access to the WoE is clean is determined by $(P_{clean\_access})^L$. Therefore the probability of clean access in WoE for sequential and hyper branched addressing are:

$$P_{Cov}^{Seq} = \left[\sum_{i=0}^{S-1}\binom{\frac{L.A}{D}}{i}.P_{FG}^{i}.(1-P_{FG})^{\frac{L.A}{D}-i-1}\right]^L \tag{14}$$

$$P_{Cov}^{Hyper} = \left[\sum_{i=0}^{S-1}\binom{L.A}{i}.P_{FG}^{i}.(1-P_{FG})^{L.A-i-1}\right]^L \tag{15}$$

Based on this analysis it is interesting to investigate how the program behavior in terms of sequantiality could result in different utilization of the IDC. Figure 10 illustrates the probability of clean access to the WoE for two programs with identical length in WoE utilizing the same size IDC but one with sequential execution in the WoE, and the other with hyper branched binary addressing. One can clearly identify that a program with sequential binary addressing maintains perfectly clean access even for $P_f > 10^{-4}$ (and therefore could go to lower Vdd) while a program with hyper branched binary addressing starts to deteriorate much earlier (i.e. at higher Vdd). For practical benchmarks usually the probability of clean access is between these two extremes.

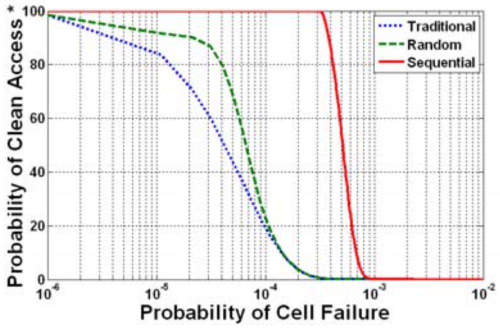

Figure 10: Probability of clean access for a complete sequential and hyper branched access to the cache when IDC structure is used, compared with Probability of clean access for the cache of the same size when IDC is not in use.

Figure 11 reveals the effect of changing the size or associatively of the IDC on the probability of clean access. The WoE in all cases is fixed and a sequential binary addressing in the WoE is considered. From Figure 11 it is concluded that increasing the associativity results in better coverage. However increasing associativity results in increased dynamic power consumption due to the increase in the number of active comparisons in each access. On the other hand, increasing associativity, potentially reduces soft misses (due to access to a defective location) and therefore total energy consumption could improve, which implies an optimal point for the associativity of the IDC. In other words, there exists a pair of (Associativity, #of Rows) for each program that will result in the lowest power consumption. However, since for most programs WoE behavior changes dynamically and different programs have different size WoEs it is not possible to find one optimal solution for all scenarios. However, the system can dynamically change the size and associativity of the IDC to target this optimal operating point.

Another interesting observation is the relation between the size of the WoE and probability of clean access, or the ability of the IDC to provide a defect free view of the WoE for the processor. Figure 12 illustrates this relationship for a sequential program.

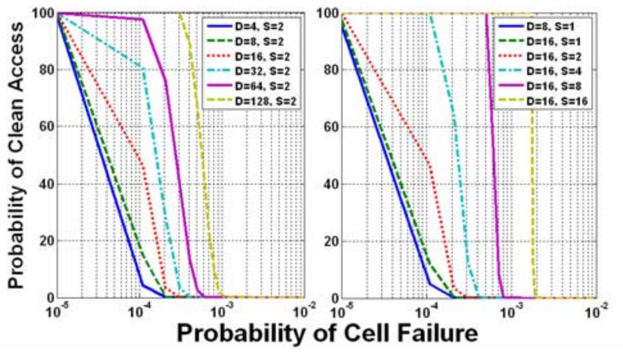

Figure 11: Comparison of Probability of clean access when changing the cache size and associativity

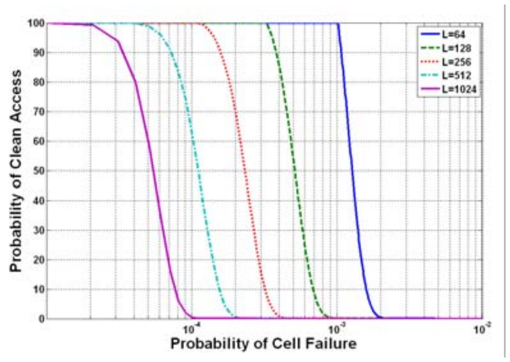

Figure 12: Probability of clean access for different sizes of WoE

VIII. USING IDC TO REALIZE A FASTER MACHINE

Basically IDC increases the fault tolerance of the Cache from that handled by redundancy alone to a much higher failure rate. On the other hand based on what was discussed before regarding the distribution of access and write time, we have demonstrated that by shortening the cycle time the probability of failure increases. An alternative way to utilize the IDC is to realize the fastest fault-free machine at a given supply voltage by shortening the cycle time up to the limit of IDC fault handling capability. This effectively defines the upper limit of a VFS policy at each voltage point.

In order to demonstrate the idea for the system defined in Table 5 and using SimpleScalar[5] we simulated the system for different random distribution of faults in the L1 data and instruction caches. As it turned out for probability of failure of $6x10^{-4}$ or less the IDC is capable of hiding the defect from the processor for all the SPEC2000 benchmarks. Knowing this, we searched at each voltage point (trough a MonteCarlo simulation) for an access time that provides such probability of failure.

In order to define a baseline, the same analysis is performed for a cache without IDC. In this case, the cycle time should be large enough such that the resulting probability of failure achieves the production yield expectation, meaning it is small enough that it could be tolerated using available redundancy. Consider, as an example, that the probability of failure of 10^{-6} is desired to meet the expected yield for a 16KB cache. Through a Monte Carlo simulation for a 10^{-6} failure rate, one can obtain the associated cycle time. Figure 13 compares the minimum cycle time of a 16KB IDC to that of a traditional cache of the same size. As illustrated, the IDC allows the system to operate at lower cycle times that are can be up to 25% shorter at sub-500mV supply.

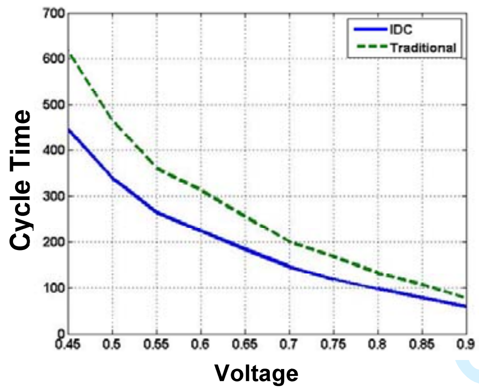

Figure 13: Comparison of Probability of clean access when changing the cache size and associativity

IX. Simulation Results

A. *Simulation setup*

We used SimpleScalar [5] as a system simulator to model and study how using an IDC improves MILPV tolerance and how this added tolerance is translated into energy savings. SimpleScalar is augmented with necessary extensions to model an IDC-enabled cache hierarchy. In order to obtain energy consumption metrics SimpleScalar is also enriched with bookkeeping code to calculate the energy consumption based on a per access scenario. In order to obtain a figure for dynamic energy consumption of IDC, Cache and BLB, we used post-layout energy consumption figures for 32nm technology. The simulated system configuration data is provided in Table 5.

Table5: SimpleScalar simulation setup

Variable	Value
Issue/decode	*4 words*
Branch prediction	*Combination*
ROB/LSQ	*128/64 entry*
L1 data cache	*16KB 3cycle pipelined, 4bank, 4 ways*
L1 inst cache	*16KB 3cycle pipelined, 4bank, 4 ways*
L2 cache	*Unified, 1M, 4way, 16cycle, 4 way*
Memory access latency	*(400 +8 per 8bytes) cycles*
L1 IDC	16 rows, 2ways
L2 IDC	64 rows,2 ways

B. *Benchmark simulation results*

The cycle time of the cache structures L1 and L2, based on Equation (2) is defined using parameters (a, b, c) = (1.5, 0, 60) at each voltage point. The associated probability of cell failure curve for this simulation is drawn in Figure 2. In order to calculate the energy consumptions, based on failure probability curves, we assumed uniformly distributed defects in cache structures. After randomly fast forwarding through 1B to 2B instructions in the benchmark, we ran the benchmark for a total of 1B instructions, calculating the sum of dynamic and leakage energy consumption. The process of distribution of defects and the benchmark simulation is repeated 250 times to reduce the chances of accidental over or under utilization.

For our case study, we consider the total energy consumption in L1 data and instruction caches as well as the IDC of data and instruction cache and their associated defect map. Figures 14 and 15 illustrate the percentage of energy saving of each benchmark in data and instruction cache respectively when compared to its total energy consumption at the nominal voltage. It is assumed that the IDC and BLB are power gated when the whole cache is operating at nominal voltage in order to eliminate their leakage.

The savings in dynamic energy consumption increase when we reduce the supply voltage but only up to a certain point. We will refer to this point as the "turning point", or V_{TP}. Further lowering the $V_{dd} < V_{TP}$ increases the number of defective FGs beyond the tolerance of the proposed architecture and results in an abrupt increase in miss rate, and as a result a sharp increase in dynamic energy consumption. In addition to the voltage level, particular execution properties of each benchmark affect the extent of their energy saving. As stated previously, the IDC is much smaller than the total number of defective words in the cache and therefore cannot store all the instructions that are mapped to defective FGs in the cache. Instead, the IDC only stores those defective words that fall inside the WoE. When the WoE changes, the IDC should be updated with instructions mapped to defective FGs in the new WoE. Based on this discussion one infers that programs whose WoE is rapidly changing (multiple branching, and generally less local), or those that have a very large window (such that the number of defective words inside that window exceeds the IDC size), could not fully benefit from the proposed architecture. Such programs will result in lower energy savings. The "eon" and "crafty" benchmark in Figures 14 and 15 are programs with multiple small but long executing loops, small execution window and good locality..

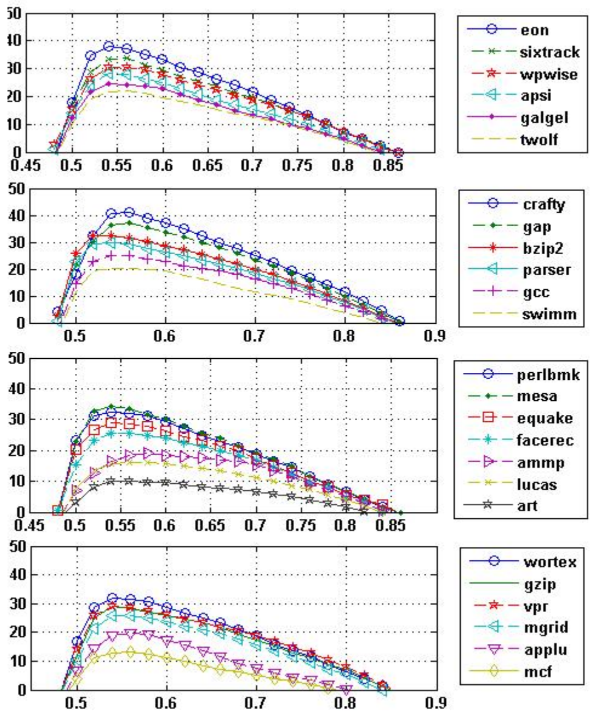

Figure 14: Percentage saving in total energy consumption for different SPEC2000 benchmarks in L1 data cache

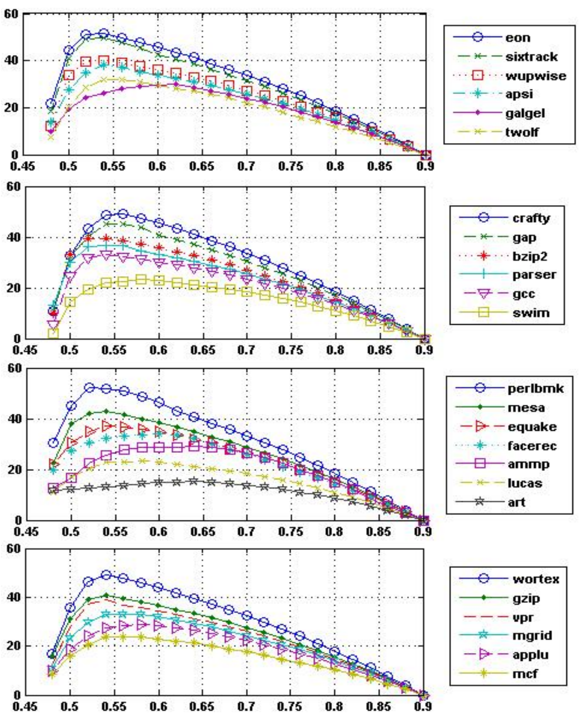

Figure 15: Percentage saving in total energy consumption for different SPEC2000 benchmarks in L1 instruction Cache

On the other extreme, "mcf" and "art" benchmark have fewer localities and rapidly changing execution window. The other benchmarks fall between these two extremes. Note that when migrating to more aggressive nodes (smaller geometries), leakage will become dominant, allowing lower V_{TP}. However, there will always be a saturation point since the extra soft misses not only increase the dynamic power consumption by requiring access to lower level memories, but also increase the execution time allowing the circuit to leak for a longer period.

The energy savings that are reported in Figure 14 and 15 are for a VFS policy defined by parameters (a,b,c) = (1.5,0,60). By choosing a different set of parameters for (a,b,c) the failure rate and therefore the amount of utilization of IDC at different voltages will change. Table 6 reports the best, worst and average power savings as well as V_{TP} associated with different policies (32 nm technology). As a reminder, when a and b are chosen 0, the voltage scaling policy is FFVS.

C. *Choosing the IDC size*

By choosing different sizes of IDC, one could trade area for larger fault tolerance and consequently change the amount of expected power savings. Using a larger size IDC reduces the number of soft misses but at the same time increases static power. Depending on its organization, a larger IDC could also consume larger dynamic energy. We now attempt to quantify those tradeoffs using area and power savings estimates based on estimation and simulation approach.

Table 6: Energy savings for benchmarks with minimum and maximum voltage turning points and average energy saving for all benchmark at average turning point in 32 nm technology

Policy, (a,b,c)	Best energy savings (%), $V_{TP}(V)$	Worse energy savings (%), $V_{TP}(V)$	Average energy savings (%), $V_{TP}(V)$
VFS(0.5,2,50)	36.70, 0.74	18.61, 0.83	27.11, 0.8
VFS(0,2,100)	31.14, 0.6	18.21, 0.69	24.64, 0.65
VFS(0,2,100)	48.21,0.52	18.54,0.63	33.92,0.53
FFVS(0,0,180)	26.92,0.65	16.13,0.69	21.71, 0.67
FFVS(0,0,220)	36.56,0.6	21.15,0.64	30.01,0.62
FFVS(0,0,260)	44.16,0.54	21.63,0.56	32.12,0.55

Area overhead analysis:

Using area figures from CACTI for the main cache (16KB, 2 Banks), IDC (of size 32 rows /2 way associative) and BLB in 32 nm technology, an area overhead of 7.69% is incurred. This area overhead is less than that reported in [3] since the defect map is generated at a larger granularity (FG rather than word) and therefore the area of BLB is reduced. Our layout of an IDC enabled cache (in 45nm technology using

TSMC-45LG and ICstudios) resulted in a design with 7.31% area overhead. The total area penalty (including BLB and IDC) for different choices of IDC size in 32nm technology node is summarized in Table 7.

Note that based on the size of the cache, the technology size that the structure is implemented at, the ratio of the size of IDC to the size of the cache and many other parameters, this area overhead is subject to change. However our purpose in providing the area overhead data is to clarify that the area cost to implement the proposed cache is indeed small and acceptable. Furthermore, it is important to note that the proposed scheme can also be used to correct for manufacturing defects and therefore remove the need for Built-In Self Repair (BISR) (but not BIST), the actual penalty can be much less when compared to memories with BISR.

Table 7: Total Area penalty of L1 Instruction Cache when using different IDC sizes

IDC Size	% Area Penalty
4 row, 2 way	4.81
8 row, 2 way	5.77
16 row, 2 way	7.69
32 row, 2 way	11.54
64 row, 2 way	19.24

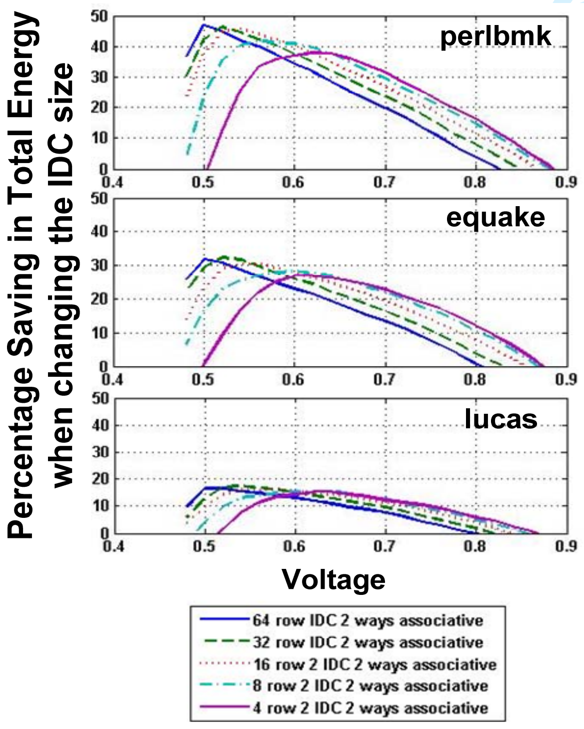

Figure 16: Percentage saving in total energy consumption for perlbmk, equake and lucas benchmarks in L1 data cache when changing the IDC size.

Power saving analysis:

In order to illustrate how choosing the IDC size changes the expected power saving, we simulated the configuration given in Table 5 for IDC sizes of (4, 8, 16, 32 & 64) across 3 different benchmarks. Benchmark 'perlbmk' represent the highly localized benchmark that well utilizes the IDC. On the other hand benchmark 'lucas' represent the opposite extreme benchmarks with less localized addressing pattern and finally benchmark 'equake' that represent the average case benchmarks.

Figure 16 illustrates the energy savings for the above benchmarks as a function of V_{dd} and IDC size. There are few interesting points to note in this figure. The first noticeable trend is the difference in the expected energy savings of each bench mark for different sizes of IDC at lower voltages. At lower voltages, the smaller IDCs consume less static power and therefore result in better energy savings. However the smaller size IDCs reach the turning point earlier than larger size IDCs; therefore their maximum power savings could be less than that of a larger IDCs. As the IDC size grows, the maximum power saving also grows but the difference with the previous point become smaller, and at some point (IDC size =64 in average and low benchmark) even the larger size IDC is expected to have equal or less power saving compare to a smaller IDC size (due to large increase in the probability of failure at lower voltages and also increased static power saving in the larger size IDCs).

When choosing the IDC size, area penalty should also be considered. As illustrated in Table 7, the area penalty for IDC size of 32 and 64 are 11.54% add 19.24% accordingly which might make the architecture less favorable due to cost. For the case of our simulation as noted in Table 5, we choose the IDC size of 16 for both instruction and data cache for an area penalty of 7.69% which provides substantial power savings with reasonable area overhead.

X. CONCLUSION

In this paper, we targeted caches for aggressive supply voltage scaling. We present a new architecture that uses an auxiliary cache (IDC) and a defect map (BLB) to enable significant overall power savings while maintaining low failure rates. The proposed approach can be extended in many ways, such as considering whole memory hierarchies, and defining policies for voltage-frequency power management that are more effective than traditional ones assuming "defect free" operation. The impact of this technique becomes more pronounced with decreased process geometries.

References

[1] S. Mukhopadhyay, H. Mahmoodi, K. Roy, "Modeling of failure probability and statistical design of SRAM array for yield enhancement in nanoscaled CMOS," Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on , vol.24, no.12, pp. 1859-1880, Dec. 2005

[2] A. K. Djahromi, A. Eltawil, F. J. Kurdahi, R. Kanj, "Cross Layer Error Exploitation for Aggressive Voltage Scaling," Quality Electronic Design, 2007. ISQED '07. 8th International Symposium on , vol., no., pp.192-197, 26-28 March 2007

[3] M. A. Makhzan (A. Sasan), A. Khajeh, A. Eltawil, F. J. Kurdahi, "Limits on voltage scaling for caches utilizing fault tolerant techniques," Computer Design, 2007. ICCD 2007. 25th International Conference on , vol., no., pp.488-495, 7-10 Oct. 2007

[4] http://www.hpl.hp.com/personal/Norman_Jouppi/cacti4.html

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

[6] P. Shirvani, E. J. McCluskey, "PADded cache: a new fault-tolerance technique for cache memories ," VLSI Test Symposium, 1999. Proceedings. 17th IEEE , vol., no., pp.440-445, 1999

[7] H. T. Vergos, D. Nikolos, “Efficient Fault Tolerant Cache Memory Design”, Micro processing and Microprogramming Journal, vol.41, no.2, pp.153-169, 1995.

[8] G. S. Sohi, "Cache memory organization to enhance the yield of high performance VLSI processors," Computers, IEEE Transactions on , vol.38, no.4, pp.484-492, Apr 1989

[9] A. F. Pour, M. D. Hill, "Performance implications of tolerating cache faults," Computers, IEEE Transactions on , vol.42, no.3, pp.257-267, Mar 1993

[10] L. Xiao, J. C. Muzio, "A fault-tolerant multiprocessor cache memory," Memory Technology, Design and Testing, 1994., Records of the IEEE International Workshop on , vol., no., pp.52-57, 8-9 Aug 1994

[11] Y. Ooi, M. Kashimura, H. Takeuchi, E. Kawamura, "Fault-tolerant architecture in a cache memory control LSI," Solid-State Circuits, IEEE Journal of , vol.27, no.4, pp.507-514, Apr 1992

[12] A. Agarwal, B. C. Paul, S. Mukhopadhyay, K. Roy, "Process variation in embedded memories: failure analysis and variation aware architecture," Solid-State Circuits, IEEE Journal of , vol.40, no.9, pp. 1804-1814, Sept. 2005

[13] C. Wilkerson, G. Hongliang, A. R. Alameldeen, Z. Chishti, M. Khellah, L. Shih-Lien, "Trading off Cache Capacity for Reliability to Enable Low Voltage Operation," Computer Architecture, 2008. ISCA '08. 35th International Symposium on , vol., no., pp.203-214, 21-25 June 2008

[14] A. J. Bhavnagarwala, T. Xinghai, J. D. Meindl, "The impact of intrinsic device fluctuations on CMOS SRAM cell stability," Solid-State Circuits, IEEE Journal of , vol.36, no.4, pp.658-665, Apr 2001

[15] http://www.eas.asu.edu/~ptm

[16] A. Agarwal, K. Roy, T. N. Vijaykumar, "Exploring high bandwidth pipelined cache architecture for scaled technology," Design, Automation and Test in Europe Conference and Exhibition, 2003 , vol., no., pp. 778-783, 2003

[17] M. A. Makhzan (A. Sasan), A. Khajeh, A. Eltawil, F. J. Kurdahi, "A Low Power JPEG2000 Encoder With Iterative and Fault Tolerant Error Concealment," Very Large Scale Integration (VLSI) Systems, IEEE Transactions on , vol.17, no.6, pp.827-837, June 2009

[18] K. Flautner, K. Nam Sung, S. Martin, D. Blaauw, T. Mudge, "Drowsy caches: simple techniques for reducing leakage power," Computer Architecture, 2002. Proceedings. 29th Annual International Symposium on , vol., no., pp.148-157, 2002

[19] H. Homayoun, A. Sasan (M.A. Makhzan), A.V. Veidenbaum, “Multiple sleep mode leakage control for cache peripheral circuits in embedded processors” CASES 2008

[20] M. A. Lucente, C. H. Harris, R. M. Muir, "Memory system reliability improvement through associative cache redundancy," Custom Integrated Circuits Conference, 1990., Proceedings of the IEEE 1990 , vol., no., pp.19.6/1-19.6/4, 13-16 May 1990

[21] Z. Bo, D. Blaauw, D. Sylvester, K. Flautner, "The limit of dynamic voltage scaling and insomniac dynamic voltage scaling," Very Large Scale Integration (VLSI) Systems,

[22] Xinghai Tang; De, V.K.; Meindl, J.D., "Intrinsic MOSFET parameter fluctuations due to random dopant placement," *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on* , vol.5, no.4, pp.369-376, Dec 1997

[23] K. Ishimaru "45nm/32nm CMOS challenge and perspective," Solid State Device Research Conference, 2007. ESSDERC 2007. 37th European , vol., no., pp.32-35, 11-13 Sept. 2007.

[24] Krishnarnurthy, R.K.; Alvandpour, A.; De, V.; Borkar, S., "High-performance and low-power challenges for sub-70 nm microprocessor circuits," *Custom Integrated Circuits Conference, 2002. Proceedings of the IEEE 2002* , vol., no., pp. 125-128, 2002.

Introduction:

The authors would like to thank the reviewers for their helpful suggestions. We have revised the manuscript and applied changes as suggested by the reviewers. While the entire manuscript has been reviewed and modified, sections that have been significantly re-worked are written in a blue font to ease the review process. The authors attempted -as best possible- to strike a balance between conciseness and results reporting. The aim is to create an informative paper for the reader that clearly articulates the key contributions.

Notes:

1) Note that any references indicated by (ref) are related to this document and those indicated by [ref] are related to the manuscript.
2) Note that reviewers' comments are in *italics* and the response is in regular font.

SPECIFIC FEEDBACK TO AUTHORS

Response to Reviewer #1:

1- *Some figures and its text being used are too small. Too long. To fix the above issue, extra pages will be needed.*

Those figures have been reworked and resized

2- *No implementation results in no data for area penalty.*

Area penalty for a case study of 16KB cache is discussed in section VIII.C.

3- *The biggest Cache issue is increasing Vdd_min limited by Read/Write Stability. However, only limitation by access time is discussed.*

The limitation by three type of failure (Access Failure, Read Failure & Write Failure) is discussed. Section 3 has been modified to reflect our testing methodology for these three failure types. The failure probability curves that are presented are the accumulated sum of read, write and access failures.
After marching test for each voltage the cache organization adapts itself with the chip specific defect distribution and performs the defect masking regardless of the nature or the origin of the defect.

We thank the reviewer for this insightful comment that resulted in a greatly improved presentation of the work.

4- *No assumption for each variation (How much variations of Vt,Tox,..) is described. Architectures proposed in this paper looks non innovative.*

Adopting from (1)(2)(3)(4) and most other paper in this field, variation is lumped into Vth variation to make testing and analysis practical. Adopting from (5) the The σ_{Vth} is expected to be ~ 34mV.

5- *If it would be clearly shown that how much Vdd_min can be reduced and how large area penalty is needed, this paper becomes much better.*

The area penalty and power savings are detailed in section VIII part C, furthermore, Table 7 indicates the area overhead associated with different IDC sizes. Finally Figure 16 indicates the Vdd_min for different benchmarks tested.

Response to Reviewer #2:

1- *The paper didn't introduce the proposed technique tell page 6.*
While this is true, we present an in depth discussion up front of the tradeoffs involved with defining different voltage frequency scaling (VFS) policies. Equation 2 presents a general framework that covers all possible VFS schemes of which FFVS is a special case. The authors believe that this is important to both describe and motivate the architecture.

2- *Page 5 line 6-7: Large access time makes memory tolerate process variation. This is only true for access failures. Both SNM and write margin failure are not function of access time.*

The fault behavior of three type of failure (Access Failure, Read Failure & Write Failure) is discussed. Section 3 is modified to reflect our testing methodology for these three failure types. The failure probability curves that are presented are the accumulated sum of read, write and access failures. In this work, a fault regardless of its origin is considered as a defect and should be masked.
We thank the reviewer for this insightful comment that resulted in a greatly improved presentation of the work.

3- *See US patent 5835504 by Balkin et al from IBM.*

US Patent 5835504.

a. This patent provides a tolerance mechanism in addressing part of the processor, in comparison our work doesn't touch the processor design and modify the cache organization
b. In referred patent the register addresses will be fixed pointed to point to the defective addresses, and therefore the number of registers available is equal to the number of existing bad addresses, therefore the coverage is limited, where as in our work the addresses are dynamically changes and only the defective addresses within the window of execution are mapped to the IDC. Therefore it provides much larger coverage.

4- *What if the number of defects is more than the defect map size?*

The defect map size is proportional to the size of the cache and therefore never the number of defects exceeds the defect map size. In this work for each Fetch Group (4 Words in cache) there is one bit in the defect map. We have added a statement to that effect in Section V.A.

5- *Bank redundancy and memory mapping is well known techniques in the industry, author listed many of the references. The introduction of the redundant block to operate at different voltage than normal memory seem novel to me. There are many issues that need to be worked out and solved for this proposal to work.*

We hope the explanations provided to the reviewer will help build a case that this approach is viable.

6- How is the BLB get updated?

Generating and Updating BLB is discussed in section V.A and V.B. Updating the BLB is discussed when there is a change in voltage or upon discovery of a new defect.

7- How is the system going to handle soft-miss?

Handling Soft Misses is discussed in section VI.B paragraph 1.

Response to Reviewer #3

1- In this submission, the authors have proposed a fault tolerant cache organization to cope with cache defects due to process variation. The authors have addressed the topic in terms of improvement of memory failure rate, total cache energy consumption as well as the analysis of the theoretical bound of the IDC coverage.

Given the above efforts, the current submission misses the key design methodology for the proposed cache organization that would differentiate it from the authors' previous work [3].

Current submission complete all the discussions provided in ([3] of the manuscript and (6) in the references to this document) and in addition:

a. We further explore the usage of the IDC to associative caches with multiword fetch groups. This is very important as the area overhead is much lower with this new architecture.
b. We introduce a mathematical upper bound on defect coverage of the proposed architecture relating the area and power of the proposed architecture to achievable V_{cc}^{mim} .
c. In [3] we investigated the fault coverage and voltage scalability of the system when a Fixed Frequency Voltage Scaling (FFVS) policy for scaling the cache supplied voltage is used. In this work, we extend our study and results examining the utilization of the proposed architecture for both Voltage Frequency Scaling (VFS) and FFVS policies.
d. In this work we introduce the usage of IDC as mean of gaining the fastest fault free machine at a given voltage.
e. In [3] for obtaining the power and leakage figures we utilized CACTI [4] power and leakage estimations, however in this work, in order to improve the accuracy of our results, the leakage and power figures are replaced with those obtained from post layout simulation of the cache layout.
f. The simulation environment (DINERO IV) presented in [3] has been replaced with SimpleScalar [5] to allow simulation of the architecture for well known SPEC2000 bench marks.

We have added a paragraph in the introduction that clearly describes these facts.

2- *[Page 2, section prior work] how does the proposed work compare to the one in [12]? It seems that work in [12] would also solve the cache defect issue induced by process variation. Does it imply more area/energy overhead compared to the proposed work?*

When compared with work in [12]:

a. Work in [12] down size the cache, while our methodology doesn't. For the range of voltage at which the IDC operate, the cache in [12] will be of 20 to 30% of its original size.

b. Work in [12] is capable of tolerating much lower defect rate and will be unusable if there are more defective blocks in one line compare to healthy blocks.

c. There is a logical flaw in work in [12]. For the work in [12], if blocks A and B are sequential and Block A is defective, then it is mapped to block B, consider accessing Block A and B in a close loop, it will always result in a miss, because each time it will have the data of the previously referenced block.

3- *Although the IDC coverage has been discussed, it alone is not enough to help determine the size of the IDC and BLB. This important information is absent throughout the submission.*

a. The defect map (BLB) size is proportional to the size of the cache and therefore never the number of defects exceeds the defect map size. In this work for each Fetch Group (4 Words in cache) there is one bit in the defect map.

b. The Size of IDC is obtained via simulation for different sizes of IDC. In addition the Area penalty has to be considered. Choosing the proper size of IDC is done by exploring the trade-off between area, power and performance gain and penalty. Choice of typical bench marks, Venue, marketing, cost, etc. will be affecting in the choice of final IDC size.

4- *[Page 3, Section B] the authors assumed to have all the variations lumped into a Gaussian distribution of device Vth. This is not entirely true since the mobility has different temperature dependence compared to Vth. Plus, oxide thickness is also a random variable. Have the author done any comparison between their assumption and the reality? In addition, are those variations combined still follow Gaussian distribution as the authors assumed?*

[Page 3, Section B] The authors have claimed that the cell access time follows a Gaussian-like distribution. This is not entirely true at low Vdd (<0.6V) due to other second order effects. Would it be possible to have the actual measurement distribution plotted?

Adopting from (1)(2)(3)(4) and most other paper in this field, variation is lumped into Vth variation to make testing and analysis practical. Adopting from (5) the The σ_{Vth} is expected to be ~ 34mV

5- *[Page 4, Figure 1] SM should increase when going from 0.7v Vdd to 0.9v Vdd.*

There was a typo in figure 1, it is deleted and the concept is explained in the text.

6- *[Page 4] How would the proposed organization affect the optimal selection of a,b,c in the VFS policy? Is there a formal way to do that?*

The purpose of choosing different a,b,c is to show that upon choosing different policies, different probability of failures are obtained. Section VIII explains how a voltage Scaling policy could be obtained to get maximum performance from IDC, this VFS is the upper band for all other voltage frequency scaling policies.

7- *[Page 5, last paragraph on the right column] Latches in the pipeline always work at high Vdd. Do they implicitly include level shifters? If not, how to deal with the issue when latch input is at low Vdd?*

The latches are considered to internally perform level shifting. See for example the discussion in Rabaey's book on VLSI.

8- *[Page 6, Table 3] unit for Tag stage delay should be ps instead of ns.*

The tag of Table 3 is fixed.

9- *[Page 6, Section V] As IDC and BLB are always at high Vdd, they need separate power routing. How does this contribute to the area overhead? Also, since they might be in different voltage domain of the cache, is level shifting taking into account?*

The area overhead is given in section IX.C. It includes level shifting and multiple power domains.

10- *[Page 9, Section C] Gate leakage is another contributor to the total leakage below 45nm.*
We commented on this in the paper on page 9 as follows:
"While reducing the voltage results in an exponential reduction in the sub-threshold leakage current principally due to the Drain Induced Barrier Lowering (DIBL) effect, it also results in a substantial reduction in gate leakage due to a reduction in the number of tunneling electrons [24]."

11- *[Page 11, Section VII] Is it possible to derive a generic design methodology to determine the optimal size of the IDC/BLB from the coverage/energy analysis?*

Size of BLB is know and fixed, but Generating a design methodology for determination of the size of IDC is as difficult as generating a design methodology for determining the size of the cache. Considering program behavior and randomness, and different amount of temporal and special localities, just like caches the optimum size is obtained via system simulation.

Minor comments.

** Figure 1 is neither properly indexed nor properly described in the text. Please indicate clearly what do the top and bottom graphs in Figure 1 represent. Mark the units for X and Y axes.*

This has been fixed.

** There are punctuation and spelling mistakes every now and then in the paper. Please go over it carefully.*

We have gone through the paper fixing those mistakes and hope the reviewer feels more satisfied with those fixes.

Response to Reviewer #4

1. While the problem of increasing manufacturing-induced fault tolerance in cache architectures for caches employing frequency/voltage scaling for energy savings is an interesting academic exercise, I am concerned about the practical relevance of this work because of two reasons:

(a) In most of the high performance architectures voltage supply scaling is not employed for cache memories. Can you give some examples of the commercial designs that employ voltage scaling?

Like many other organization and design methodology size as drowsy cache, the idea is not explored by industry at the present time, but is still an active area of research. The subject targets a future problem of industry and there has not been a real need in the industry to explore such degree of fault tolerance when improving the fabrication methodology has been a simpler, safer and economically better approach. However non existence of such methodology doesn't reduce its value. Academia is supposed to lead the industrial world in terms of innovation. We point out that there is significant on-going work at Intel [12] which promotes a similar paradigm of voltage scaling to caches.

(b) Your solution is based on two auxiliary structures namely BLB and IDC which are kept at a higher voltage. Do you think it is at all practical to have two voltage regulators dedicated to cache itself? In the present day designs this is not going to happen.

We agree that present designs may not have a generous supply of supply voltages. However, voltage islands are becoming an integral part of today's SoC's. Indeed, TSMC's reference flows have incorporated that since version 6.0 (we are at 10.0 today). Synopsys presentations often include adaptive voltage scaling on a per-core basis as the most potent means of reducing power (see Tom Williams' presentation on low power, for example).

It is not uncommon to see memories with multiple voltages. Memories actually do have two supplies, one for the peripherals and another for the array. Indeed, vendors of memory compilers go beyond that, providing in-core voltage step-down regulators to produce substrate biasing voltage to put memories in low-leakage state (see Virage Logic for example).

In summary, we expect a trend towards a larger variety of supply voltages on chip as the only means of combating the exponentially increasing power density.

2. What does table 2 indicate? Mean access time depends upon the distribution obtained at a given voltage point. SM and hence, the access time (frequency of operation) should be a function of mean access time and not vice-versa. However, in this table you show mean access time to change with the choice of SM. Please explain.

There was a typo in table 2 legend. The provided numbers are cycle time and not safety margin.

3. In the experimental results, only the energy aspect of the proposed cache architecture is discussed. Can this cache absorb extent of access time variation that could be introduced by process variations and difference in energy consumption can be directly compared? How is yield performance trade-off affected of the new cache compares with that of the old one. Also, different cache instances are affected differently by manufacturing-induced process variations and hence, will vary in how much energy savings are achieved. Is it not clear that which point in the energy spread does your results correspond to. Any architecture that employs voltage scaling will lead to energy savings! Can you comment on that?

Again in the results section, The results do not evaluate the fault tolerant caches very comprehensively. While the energy aspect is presented in the results, yield/performance trade-off which was discussed early on in the paper is completely ignored.

In terms of yield this cache organization provides much higher yield compare to traditional architecture because as it was referred in the text, IDC could be used for coverage of manufacturing defects as well as process variation defects. Therefore even when the static redundancy is falling short of remapping or covering all the defects in the cache, they can be left to be handled by IDC by expecting less energy savings. However in this paper the energy aspect of using IDC is explored and simulation represent an average case energy gain.

Every cache organization for having a different spread of fault distribution obtain different amount of final energy saving. The provided results are averaged over 250 different simulations each 1 billion instruction long and are expected to present an average case. (section IX.B)

[1] S. Mukhopadhyay, H. Mahmoodi, K. Roy, "Modeling of failure probability and statistical design of SRAM array for yield enhancement in nanoscaled CMOS," Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on , vol.24, no.12, pp. 1859-1880, Dec. 2005
[2] A. K. Djahromi, A. Eltawil, F. J. Kurdahi, R. Kanj, "Cross Layer Error Exploitation for Aggressive Voltage Scaling," Quality Electronic Design, 2007. ISQED '07. 8th International Symposium on , vol., no., pp.192-197, 26-28 March 2007
[3] A. Agarwal, B. C. Paul, S. Mukhopadhyay, K. Roy, "Process variation in embedded memories: failure analysis and variation aware architecture," Solid-State Circuits, IEEE Journal of , vol.40, no.9, pp. 1804-1814, Sept. 2005
[4] C. Wilkerson, G. Hongliang, A. R. Alameldeen, Z. Chishti, M. Khellah, L. Shih-Lien, "Trading off Cache Capacity for Reliability to Enable Low Voltage Operation," Computer Architecture, 2008. ISCA '08. 35th International Symposium on , vol., no., pp.203-214, 21-25 June 2008
[5] K. Ishimaru "45nm/32nm CMOS ~challenge and perspective~," Solid State Device Research Conference, 2007. ESSDERC 2007. 37th European , vol., no., pp.32-35, 11-13 Sept. 2007
[6] M. A. Makhzan (A. Sasan), A. Khajeh, A. Eltawil, F. J. Kurdahi, "Limits on voltage scaling for caches utilizing fault tolerant techniques," Computer Design, 2007. ICCD 2007. 25th International Conference on , vol., no., pp.488-495, 7-10 Oct. 2007