Scaling of Architecture Level Soft Error Rate for Superscalar Processors *

Xiaodong Li[1], Sarita V. Adve
Department of Computer Science
University of Illinois at Urbana-Champaign
{xli3, sadve}@cs.uiuc.edu

Pradip Bose, Jude A. Rivers
IBM T.J. Watson Research Center
Yorktown Heights,NY 10598
{pbose, jarivers}@us.ibm.com

Abstract

Soft errors are a growing concern for processor reliability. Recent studies have motivated architecture level studies of soft errors. It has been shown that the architecture level has a large derating effect on the raw processor error rate. In this paper, we quantify the impact of technology scaling on the processor soft error rate, taking the architecture level derating effects and workload characteristics into consideration. For our evaluation, we use SoftArch to quantify the derating factor and soft error rate (SER) for different structures in a modern superscalar processor running SPEC2000 benchmarks. We compare the SERs across four different technologies ranging from 180nm to 65nm with the same microarchitecture. We find that with scaling, the derating factors for logic structures often decrease, the derating factors for storage elements remain roughly unchanged, and the FIT rate for the full processor roughly follows the trend for the raw SER of storage structures (i.e., the FIT rate.increases from 180nm to 90nm and decreases from 90nm to 65nm.)

1 Introduction

Moore's Law has brought tremendous improvement in performance and power consumption for microprocessors. However, these gains appear to face fundamental reliability challenges as CMOS technology scales into the deep submicron regime. In particular, with the continuous shrinking of feature size and decreasing of supply voltage, soft error issues have emerged as a new design challenge. Soft errors are transient errors caused by high energy particle strikes such as cosmic rays and alpha particles from IC packaging material. As opposed to hard errors, soft errors typically do not cause permanent damage to devices. However, they can be catastrophic for the correct execution of a program since they may flip the value stored in storage cells or change the value computed by logic circuits. Previous work has studied the soft error rate (SER) trends with technology scaling for different types of circuits (SRAM, latches, and logic) [4, 9, 10] and there is a growing concern about soft error reliability with future technologies.

Most research on soft errors has focused on modeling and exploring the issue at the device and circuit level. Recently, however, there has been work on understanding the impact of soft errors at the architecture or micro-architecture level [1, 5, 8, 13, 14] for several reasons. First, device or circuit level solutions are expensive in terms of hardware resource, performance, and power consumption. Effective architecture level solutions might greatly reduce the cost of the soft error protection schemes. Second, research has shown that there is a large derating effect at the architecture level. Many of the raw errors that occur at the device/circuit level may be masked at the architecture level. For example, Wang et al. report that more than 85% of the raw errors are masked at the micro-architecture level and the architecture level [13]. Understanding and quantifying this derating effect of different structures can help to identify the most vulnerable parts of the processor and design effective protection schemes.

Although the impact of technology scaling on SER for different kinds of circuits has been extensively studied, there has been no previous work studying the effect of scaling on processor SER taking architectural derating effects into consideration. In this paper, we make the first attempt to quantify the impact of technology scaling on the architecture level processor SER. For our evaluation, we use SoftArch [6], an architecture level modeling methodology and tool, to quantify the SER for a modern processor over four technology generations ranging from 180nm to 65nm.

Our contributions: This work represents the first quantitative evaluation of the impact of technology scaling on the processor SER, from the architectural perspective and taking into consideration of workload characteristics. We apply the SoftArch model to a modern superscalar processor running

*This work was supported in part by the National Science Foundation under Grant No. CCR-0313286.

[1] Xiaodong Li is a Ph.D. student at the University of Illinois at Urbana-Champaign. This work was done while he was a co-op at the IBM T.J. Watson Research Center.

SPEC2000 benchmarks. Our model incorporates most of the major storage and logic structures on chip, including the instruction buffer, instruction decode unit, instruction queues, floating point unit, fixed point unit, and TLB. We assume the caches and load/store queues are protected with ECC. We extend the SoftArch model by adding scaling specific parameters for several different CMOS technologies. In our study, we model the scaling effect of taking one processor design, and gradually scaling the chip down from 180nm to 65nm, without any modifications to the micro-architectural pipeline. Different from previous scaling studies on SER for individual types of circuits, our result shows the scaling trend of the whole processor SER which includes the effect of scaling on both the circuit SER and the architectural derating factor.

2 Background

2.1 Soft Errors

A processor consists of storage elements (including SRAM, registers) and logic structures. These exhibit different soft error characteristics.

Storage Elements: Storage elements are used to store bit value ”0”s or ”1”s in a processor. When alpha particles or neutrons hit a transistor in a storage element, the amount of charge collected by the device may exceed the critical charge Q_{crit}, which is defined as the minimal amount of charge needed to flip the element. The value stored in the storage element will be flipped, thus causing a soft error.

Logic Structures: Combinational logic in a processor is used for computation and control. When neutrons hit transistors in the logic circuit, some electrons and holes are created in the silicon. Under a certain electrical field, the charge will form a current pulse and a voltage pulse. This pulse will propagate through gates and attenuate as it goes through each gate (*electrical masking*). Some signals will be masked by the logic masking effect of the gates (*logic masking*). When the pulse finally reaches the latch, if it does not fall into the latch window which is the time window between the set up time and the hold time, the error will also be masked (*latch window masking*). Traditionally, logic SER was ignored in most architectural studies because the masking effects make logic SER much lower than the SER of storage elements. But as technology scales down, the masking effects for logic are diminishing and the logic SER is projected to increase drastically [10].

2.2 SoftArch Model

To study the effect of scaling on processor SER, we use SoftArch [6]. SoftArch is an architecture level model and tool used to quantify the soft error reliability of a processor. It is a probabilistic model that works with a high-level architectural timing simulator. As discussed in detail in [6], it is more efficient and general compared to the previous fault injection [13] and AVF methods [8]. Next, we give a short summary of the model and how SoftArch works.

SoftArch models the error generation and propagation process in the processor. It tracks the error probability of each *value* communicated or computed in the processor. In the processor, a value might be erroneous because of either error generation or propagation. If during a value’s residence time in a structure (storage element or logic), it is physically struck by a particle, a new error source will be generated. We refer to this as error *generation*. The erroneous value could also be the result of a communication of an erroneous value or might be computed using one or more erroneous input values. We refer to this as error *propagation*. In SoftArch, the error generation process is modeled with some probability. For example, for a storage element, this is modeled using the combination of residence time of the value in the storage element and the raw SER for the storage element. Error generation in logic is also modeled with a simple probability abstraction. For the error propagation process, we apply probability theory on the error probabilities of the propagation sources.

Not all wrong values cause program failures. Only the erroneous values which propagate to certain points of the processor could cause processor or program failures. Thus, during the program execution, SoftArch identifies the values that could actually affect program outcome (e.g., values sent to an output device or affecting control). Then, SoftArch uses the tracked errors and the simulator timing data to determine the possible timing of such errors and the probability for such a failure. Based on this information, we can calculate the failures in time (FIT) rate and mean time to failure (MTTF) for the processor using basic probability theory. SoftArch also keeps enough information to be able to attribute the overall FITs to different processor structures.

3 Impact of Scaling on Processor SER

Many of the raw errors that occur at the device or circuit level may be masked at the architecture level. This is the architecture level derating effect, where the derating factor is defined as the percentage of errors that do not get masked. The processor SER depends on both the raw SER and the architecture level derating factor. Thus, in order to predict the scaling trend of the processor SER, we need to incorporate the scaling trend of both the raw SER and the derating factor. Next, we discuss the factors that affect the scaling of the raw SER and the derating factor.

3.1 Impact of Scaling on Raw SER

The scaling of raw SER for storage elements and logic has been extensively studied [2, 10, 4].

SER of storage elements: As CMOS technology scales down, the following factors contribute to the scaling of the raw SER of storage elements: (1) Q_{crit} which is the minimum amount of charge to upset a device decreases with technology scaling because of the reduced capacitance at each node. (2) The lower supply voltage V_{dd} also contributes to reducing Q_{crit}. (3) The charge collection efficiency decreases which makes it harder to collect charge. In this paper, we use results from Karnik et al. [4] that predict that the SER for storage elements increases from 180nm to 90nm and decreases slightly from 90nm to 65nm.

Logic SER: As mentioned in Section 2.1, previously SER of logic was not a major concern because of several circuit level masking effects. But as technology scales down, these masking effects are diminishing and the logic SER is projected to increase drastically. The main reasons are the following: (1) Q_{crit} decreases because of the reduced node capacitance and lower supply voltage. (2) As technology scales down, pipelines will get deeper. A deeper pipeline means fewer levels of gates between latches. That increases the probability that errors make their way to the latches. (3) Higher frequency and faster clock reduce the latch window masking effect. Overall, previous studies have predicted that raw logic SER will increase by orders of magnitude and become comparable to the SER of unprotected SRAM structures [10].

3.2 Impact of Scaling on Architectural Derating

As described in [6], the derating factor for storage and logic structures mainly depends on two factors: (1) *base utilization* which is the fraction of values that are alive for storage elements and the fraction of time that the structure is used for logic structures. (2) *effective utilization* which is the percentage of the values that are read or computed from the structure and will affect the program outcome. Assuming no changes to the microarchitecture, as technology scales downward, there are mainly two factors that will lead to changes in the derating factor.

1. **Frequency:** As the processor frequency increases, the processor cycle time will decrease. Thus, the absolute time that a certain value stays in a storage element will have a tendency to decrease if the number of processor cycles stays the same.

2. **Latency of off-chip memory:** Since the processor frequency scales faster than the memory speed, the speed gap between processor and memory gets larger. As a result, the memory latency in terms of processor cycles for future generation processors will increase. Thus, intuitively, the processor would spend more cycles stalling for memory which would change the derating factor of the processor.

Technology Parameters	
Process technology	180nm
Processor frequency	1.1 GHz
Processor Parameters	
Fetch rate	8 per cycle
Retirement rate	1 dispatch-group (=5, max) per cycle
Functional units	2 Int, 2 FP, 2 Load-Store, 1 Branch
Issue queue entries	FPU 20, Load/Store/Int 36, BR 12
Integer FU latencies	1/4/35 add/multi/div (pipelined)
FP FU latencies	5 default, 28 div. (pipelined)
Register file size	80 integer, 72 FP
iTLB/dTLB entries	128/128
Memory Hierarchy Parameters	
L1 Dcache	32KB, 2-way, 128-byte line
L1 Icache	64KB, 1-way, 128-byte line
L2 (Unified)	1MB, 4-way, 128-byte line
Contentionless Memory Latencies	
L1/L2/L3 Latency	1 /10 /77 cycles

Table 1. Parameters for the base superscalar processor.

4 Experimental Methodology

4.1 Processor Modeled

We use the SoftArch tool integrated with the Turandot simulator [7]. Turandot is a trace-driven performance simulator that models the timing of the various pipeline stages of a processor in detail. The base processor we simulated is a 180nm out-of-order 8-way superscalar processor with parameters summarized in Table 1. We assume an off-chip large L3 cache with a 100% hit rate.

We model soft errors in all important structures in the processor, including the instruction buffer (IBUF), instruction decode unit (IDU), fixed-point unit (FXU), floating point unit (FPU), integer and floating point register files (REG), instruction queues (IQ), iTLB, and dTLB. We do not model soft errors in the predictor structures since these do not cause processor failures. We also do not model soft errors in the caches since caches are usually protected with ECC. We assume the Load/Store queues are also protected with ECC.

4.2 Scaling Methodology

We study the architecture level FIT rate for the modeled processor for four technology generations, ranging from 180nm to 65nm. We assume there are no modifications to the processor micro-architectural pipeline with scaling. Effectively, we scale the same chip from 180nm to 65nm technologies.

Tech gen	**Freq**	**Vdd**	**L3 Latency**	λ (FIT/bit)	FPU e_{logic}	FXU e_{logic}	IDU e_{logic}
180 nm	1.1 GHz	1.8 V	77 cycles	$5.7 * 10^{-4}$	$1.45 * 10^{-22}$	$1.06 * 10^{-22}$	$7.61 * 10^{-23}$
130 nm	1.35 GHz	1.5 V	94 cycles	$6.0 * 10^{-4}$	$9.96 * 10^{-23}$	$7.34 * 10^{-23}$	$5.25 * 10^{-23}$
90 nm	1.65 GHz	1.2 V	115 cycles	$7.4 * 10^{-4}$	$5.97 * 10^{-23}$	$4.40 * 10^{-23}$	$3.15 * 10^{-23}$
65 nm	2.0 GHz	0.9 V	140 cycles	$7.1 * 10^{-4}$	$3.26 * 10^{-23}$	$2.40 * 10^{-23}$	$1.73 * 10^{-23}$

Table 2. Scaling parameters for the simulated processor.

Table 2 summarizes the parameters that change with scaling. Although with ideal scaling, the best base frequency scaling per generation should be about 43%, it is hard to achieve the ideal frequency boosts without significant retuning all the circuit delay paths in the processor. Therefore, we conservatively assume 22% frequency scaling per generation. Since everything on chip is scaled, we assume the on-chip storage structures such as register files, instruction queues, TLBs, and caches scale linearly with the transistors and their access times *in terms of processor cycles* stay the same. For the off-chip L3 cache, we assume the *absolute* access time stays the same; therefore, its access time in terms of processor cycles increases about 22% for each generation.

Table 2 also gives the scaled values for the raw SER for storage structures, denoted λ. Additionally, for each type of logic circuit (FPU, FXU, and IDU), it gives the probability that, given correct inputs, the result produced by the circuit at the end of the corresponding computation is incorrect because of soft errors. This probability is denoted as e_{logic}. λ and e_{logic} are inputs into SoftArch [6]. We calculate them based on published literature as follows (similar to the methodology in [6]).

For storage structures, Irom et al. [3] and Swift et al. [11] report the SER cross section for the TLB and floating point registers for PowerPC processors. We assume the same values for the instruction queue, integer register file, and the instruction buffer, and calculate the raw SER [6]. We then scale the raw SER for different technologies using the scaling curve provided by Karnik et al. [4]. As shown in Table 2 the raw SER (λ) of storage elements goes up as technology scales down from 180nm to 90nm and then decreases slightly from 90nm to 65nm.

For logic structures, we determine the raw SER by using scaling data for logic chains and for latches by Shivakumar et al. [10] and estimates of gate and latch counts for each modeled structure (e.g., IDU, FPU, FXU) as follows (again, similar to [6]). First, since the base simulator parameters [7] are chosen to correspond roughly to the POWER4 microarchitecture [12], and since actual unit-wise gate/latch count information for such commercial processors is not available, we first estimated the relative areas of each modeled unit from published floorplans of the POWER4. Since the total transistor count for the processor is known, we could then assign area-based estimates of transistor counts for each modeled structure. Reasonable assumptions about transistor density differences between SRAM and logic dominated structures were also factored in. For non-logic array structures (e.g., caches) the transistor count was easier to estimate from the published sizes of these structures, and using reasonable assumptions about the number of transistors per cell. Second, for non-array logic structures (e.g. FXU), we assumed that there is a 30:70 ratio of areas covered by latches and logic respectively. This is a rule of thumb that works reasonably well across macros, technology generations, and different processor designs, as validated by discussions with actual designers. Third, we also assumed reasonable (designer-validated) average values for the number of transistors per latch and per logic gate. Also, the number of logic levels within a pipeline stage (logic chain length in [10]) was estimated form the design FO4 of the modeled processor. (The assumptions above can definitely result in significant errors in the gate and latch count estimates. However, since the actual values are not available for use in this study, we did the best we could for the purposes of illustrating our overall methodology to estimate the effect of scaling on architecture-level SER.) As shown in Table 2, the raw logic SER (e_{logic}) decreases with scaling. This is because logic error rate is dominated by latches and the SER for latches is going down.

4.3 Workload description

We report experimental results for 12 SPEC CPU2000 benchmarks including 6 integer benchmarks and 6 floating point benchmarks.

5 Results

5.1 Overall Results

Our results are presented in Figures 1 and 2. Figure 1 shows the FIT rate for the processor. Figure 1(a) shows the raw processor FIT rates which are calculated assuming that each raw error causes a program failure for the four technology generations. Figures 1(b) and (c) show the FIT rates for our SPECint and SPECfp benchmarks respectively. Each group consists of four bars which are for four technology generations starting from 180nm to 65nm. Each bar is further divided to show the contribution to the FIT rates from the different structures – register file (REG), instruction queues (IQ), data TLB (dTLB), instruction TLB (iTLB), integer functional unit (FXU), floating point unit (FPU), I-buffer (IBUF), and decode unit (IDU).

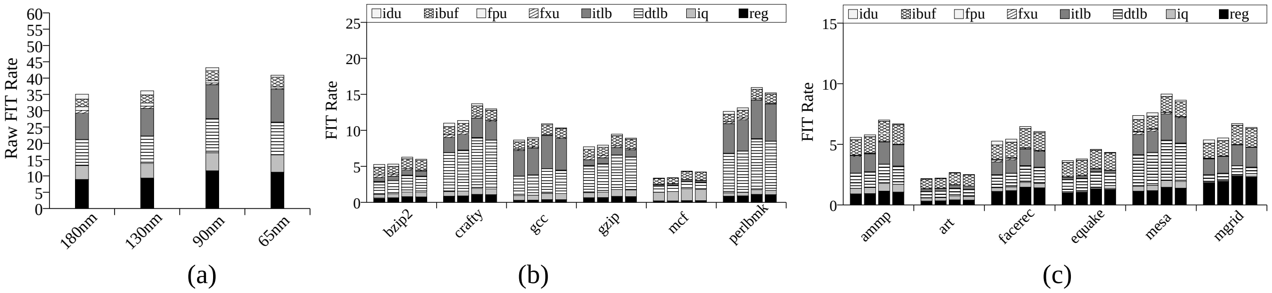


Figure 1. FIT rates (a) for raw errors, (b) with architecture masking for SPECint benchmarks, and (c) with architecture masking for SPECfp benchmarks.

Figures 2 (a) and (b) show the architectural derating factors for each structure and the entire processor for the SPECint and SPECfp benchmarks respectively for the four technology generations.

In view of the inaccuracies in our method of estimating the raw SER values (Table 2), the focus of the results presented here is not on absolute FIT rates which are almost certainly inaccurate. Instead, the goal of the ensuing analysis is to show the trends with scaling. We believe these trends are reasonably accurate.

Our high level results are the following:

FIT rate scaling: The FIT rate of the whole processor increases as technology scales from 180nm to 90nm and decreases slightly from 90nm to 65nm. The reason is that the dominating source of the FIT rate is the storage structures. The logic FIT rate is insignificant compared to the storage element FIT rate. From 180nm to 90nm, the FIT rate of storage structures increases. From 90nm to 65nm, the FIT rate of storage structures decreases slightly.

Derating factor scaling: (1) The derating factor of logic structures (FPU, FXU, IDU) decreases as technology scales down. (2) The derating factor of storage elements does not change much with technology scaling and increasing memory latency. This is the case for both SPECint and SPECfp applications.

5.2 Analysis

Next, we use a simple model to analyze the FIT rate and derating factor scaling trends. As discussed in [6], the FIT rate for a given structure is determined by three factors: *raw FIT rate for the structure, base utilization of the structure,* and *effective utilization of the structure*. The derating factor is only determined by the latter two factors. The raw FIT rate factor depends on the technology. For storage structures, the base utilization is the fraction of values that are alive. For logic structures, the base utilization is the fraction of time the structure is used. The effective utilization of a structure is the fraction of values that are read or computed from the structure that affect the program outcome.

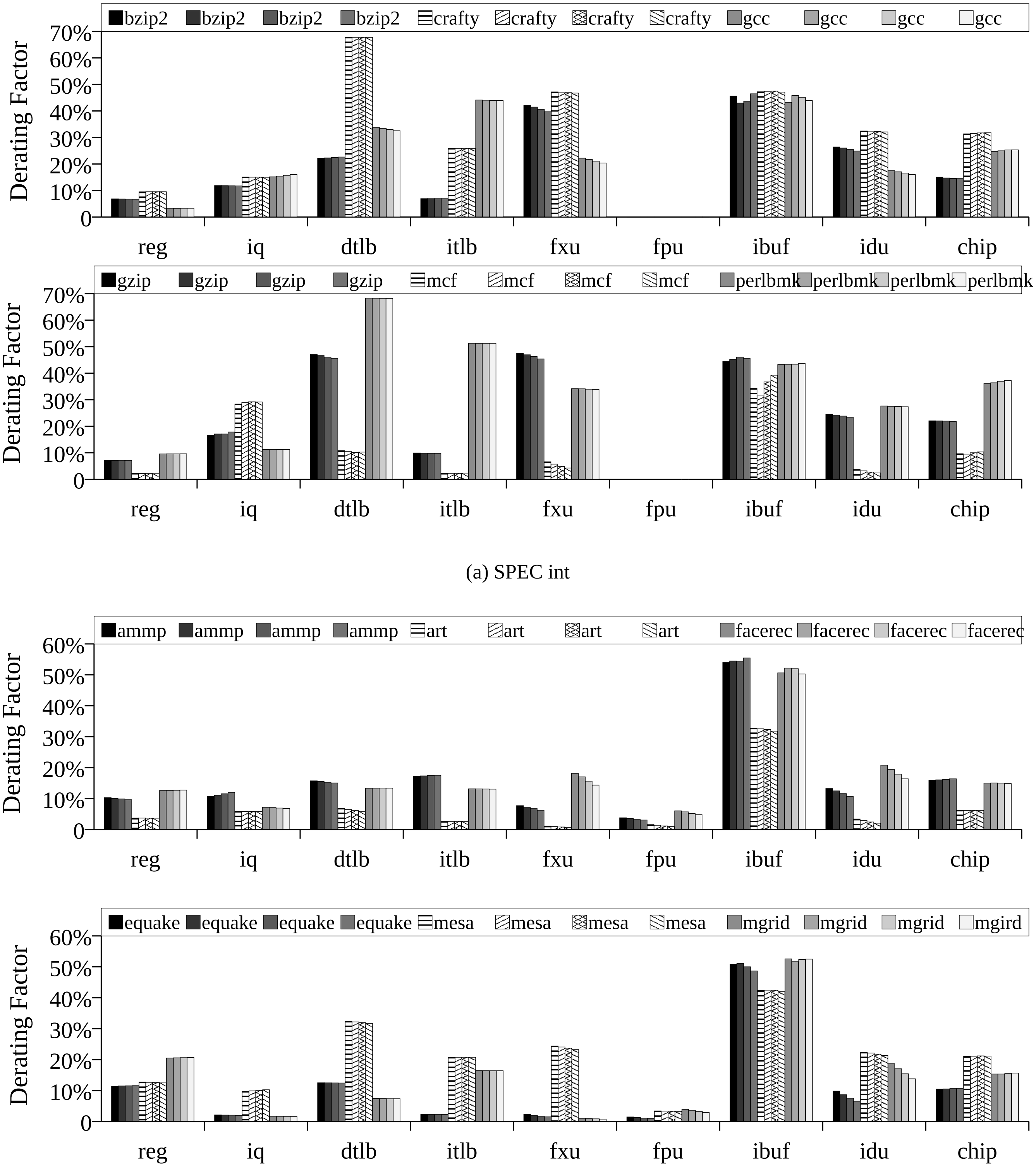
The scaling of raw FIT rate has been summarized in Section 4.2. Next we will analyze the scaling trend of the base utilization factor. It can be expressed as T_{busy}/T_{exec}. Here T_{exec} is the total execution time of the program. For logic structures, T_{busy} is the time the structure is used. For storage elements, T_{busy} is the time that the element holds live values. This is equivalent to $Cycles_{busy}/Cycles_{exec}$. Here $Cycles_{exec}$ is the number of cycles for program execution. $Cycles_{busy}$ for a logic structure is the number of cycles the structure is busy. $Cycles_{busy}$ for a storage element is the number of cycles the element has live data.

From 180nm to 65nm, the processor frequency increases 22% every generation. If there is no memory access, the value of $Cycles_{exec}$ would stay the same. But for real applications, $Cycles_{exec}$ typically increases because the increasing memory latency would delay the program execution. Figure 3 shows the increase in number of cycles for each application when the memory latency increases from 77 to 140 cycles. From Figure 3, the execution time of most integer applications is not very sensitive to the memory latency (except *mcf*), while the execution time of most floating point applications is more sensitive.

The scaling of the $Cycles_{busy}$ value is more complex. Next, we will discuss the scaling for logic and storage structures separately.

For logic structures (FPU, FXU, IDU), although the processor frequency changes, the number of committed instructions and the instruction sequence stays the same. Thus the number of operations that are critical to the outcome of the program will not change. For example, for each technology generation, there would be the same number of FPU operations and FXU operations that are critical to the program outcome. Thus, the value of $Cycles_{busy}$ would be the same for logic. As a result of the increase of $Cycles_{exec}$, the derating factor of logic would decrease.

For storage elements (reg, IQ, TLB, IBUF), $Cycles_{busy}$ is the number of cycles data is live in the element. It tends to increase with technology scaling because the memory latency gets larger from 180nm to 65nm. According to our exper-



(a) SPEC int

(b) SPEC fp

Figure 2. Architectural derating factor for each structure (a) for SPECint and (b) for SPECfp benchmarks. Note that the scales on the two graphs are different. For each application, the four bars in a graph represent the four technology generations, going from 180nm to 65nm.

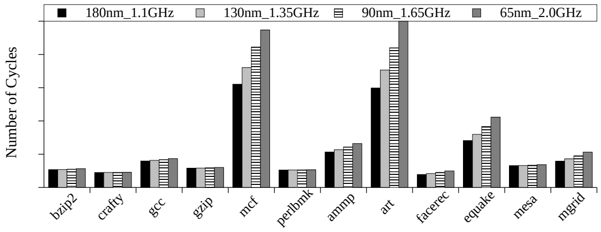


Figure 3. $Cycles_{exec}$ **for each application for different frequencies (technologies)**

iments, the rate of increase of $Cycles_{busy}$ and $Cycles_{exec}$ is similar. Therefore, the increases roughly cancel out with each other and the derating factor stays the same.

5.2.1 Summary

Based on on the scaling trend of the above factors, we explain the scaling trend of the derating factor and the FIT rate as follows:

Scaling trend of the derating factor: The derating factor depends on the utilization factor. As the technology scales down from 180nm to 65nm, the derating factor of the storage elements stays roughly the same, while for logic, the derating factor gets smaller.

Scaling trend of the FIT rate: The processor FIT rate depends on the raw FIT rates and the derating factors. For the processor and the technologies we modeled, storage elements FIT rate dominates and the logic FIT rate is insiginifant. Since the derating factor for storage elements does not change, the FIT rate of the whole processor follows the same trend as the raw FIT rate. It increases from 180nm to 90nm and decreases slightly from 90nm to 65nm.

6 Conclusions

We use SoftArch to quantify the scaling of the architecture level FIT rate and the derating factor for different structures in a superscalar processor running SPEC2000 benchmarks over four different technology generations. Our results show that as technology scales down, the derating factor for storage structures stays roughly the same, while for logic, the derating factor decreases. The processor FIT rate increases from 180nm to 90nm and then decreases slightly from 90nm to 65nm.

7 Acknowledgements

We would like to thank Pia Sanda, Lisa Spainhower, and Scott Swaney for early discussions and feedback during the course of this work and Zhigang Hu for help understanding the Turandot simulator.

References

[1] E. W. Czeck and D. Siewiorek. Effects of Transient Gate-level Faults on Program Behavior. In *Proceedings of the 1990 International Symposium on Fault-Tolerant Computing*, pages 236–243, June 1990.

[2] P. Hazucha and C. Svensson. Impact of CMOS Technology Scaling on the Atmospheric Neutron Soft Error Rate. *IEEE Transactions on Nuclear Science*, 47(6):2586–2594, Dec. 2000.

[3] F. Irom, F. F. Farmamesh, A. H. Johnson, G. M. Swift, and D. G. Millward. Single-Event Upset in Commercial Silicon-on-Insulator PowerPC Microprocessors. *IEEE Transactions on Nuclear Science*, 49(6):3148–3155, Dec. 2002.

[4] T. Karnik, P. Hazucha, and J. Patel. Characterization of Soft Errors Caused by Single Event Upsets in CMOS Processes. *IEEE Transactions on Dependable and Secure Computing*, 1(2):128–143, June 2004.

[5] S. Kim and A. K. Somani. Soft Error Sensitivity Characterization for Microprocessor Dependability Enhancement Strategy. In *Proceedings of the International Conference on Dependable Systems and Networks*, pages 416–425, Sept. 2002.

[6] X. Li, S. Adve, P. Bose, and J. Rivers. SoftArch: an Architectural Level Tool for Modeling and Analyzing Soft Errors. To appear in *Proceedings of the International Conference on Dependable Systems and Networks*, 2005.

[7] M. Moudgill et al. Environment for PowerPC Microarchitectural Exploration. In *IEEE Micro*, 1999.

[8] S. S. Mukherjee et al. A Systematic Methodology to Compute the Architectural Vulnerability Factors for a High-Performance Microprocessor. In *Proceedings of the 36th International Symposium on Microarchitecture*, 2003.

[9] H. T. Nguyen and Y. Yagil. A Systematic Approach to SER Estimation and Solutions. In *Proceedings of the 41st IEEE International Reliability Physics Symposium*, 2003.

[10] P. Shivakumar et al. Modeling the Effect of Technology Trends on the Soft Error Rate of Combinational Logic. In *Proceedings of the 2002 International Conference on Dependable Systems and Networks*, pages 389–398, June 2002.

[11] G. M. Swift et al. Single-Event Upset in the PowerPC750 Microprocessor. *IEEE Transactions on Nuclear Science*, 48(6):1822–1827, Dec. 2001.

[12] J. M. Tendler et al. POWER4 System Microarchitecture. *IBM Journal of Research and Development*, 46(1), Jan. 2002.

[13] N. Wang, T. Rafacz, J. Quek, and S. J. Patel. Characterizing the Effects of Transient Faults on a Modern High-Performance Processor Pipeline. In *Proceedings of the International Conference on Dependable Systems and Networks*, 2004.

[14] C. Weaver et al. Techniques to Reduce the Soft Error Rate of a High-Performance Microprocessor. In *Proceedings of the 31st annual international symposium on Computer architecture*, 2004.