

Multi-Bit Upsets Vulnerability Analysis of Modern Microprocessors

Athanasios Chatzidimitriou George Papadimitriou Christos Gavanas George Katsoridas Dimitris Gizopoulos
Dept. of Informatics and Telecommunications, University of Athens, Greece
{*achatz | georgepap | chrisgavanas | katsorid | dgizop*}@di.uoa.gr

Abstract—Miniaturization of integrated circuits brings more devices (thus more functionality) on the same silicon area but also makes them more vulnerable to soft (transient) errors. Assessment and understanding of the magnitude of a microprocessor’s vulnerability to soft errors in early stages of the design can steer wise, cost-effective protection decision at the hardware or software level. In recent fabrication technologies, the effect of radiation (neutrons or other particles) is significantly more severe on silicon devices and leads to increased numbers of multi-bit upsets. In this paper, we analyze the effects of multi-bit upsets in modern microprocessors, using microarchitecture level fault injection and a complete system stack. We present details about the effects of multi-bit upsets on 6 major hardware components of an ARM Cortex-A9 CPU modeled on Gem5 microarchitectural simulator, with 15 workloads across 8 fabrication technology nodes. For the purposes of our analysis, we employ and extend the GeFIN (Gem5-based Fault INjector) framework to model and analyze multi-bit faults in the hardware structures of the CPU. The enhanced version of the fault injector models multi-bit faults in adjacent areas of a structure; a very realistic case when modern silicon chips are affected by radiation. Our analysis shows that the architectural vulnerability factor (AVF) significantly increases from 1.5x (+50%) to 3.2x (+220%) between single and triple-bit faults across components. We present the aggregate multi-bit AVF of each hardware structure and each technology node from 250nm to 22nm; our results show significant AVF difference between single bit and aggregate multi-bit measurements, up to 35% as the technology node decreases – this reveals the magnitude of the assessment gap when only single bit errors are considered by any method. We report soft error Failures in Time (FIT) rates for the entire ARM Cortex-A9 CPU across technology nodes and our results show that the contribution of multi-bit upsets in the overall CPU FIT consistently increases across technologies and reaches 21% in 22nm.

Keywords—CPU reliability, soft errors, failures in time, spatial MBU, fault injection, microarchitecture simulation

I. INTRODUCTION

The constant refinement of semiconductor technologies continuously (although not always consistently) increases the density and complexity of modern microprocessor chips in favor of performance and functionality. This extreme scaling has a negative impact on the reliable operation of microprocessors, mak-

ing them more vulnerable to cosmic radiation, latent manufacturing defects, device and packaging degradation and low voltage operation [13] [14] [15] [16]. As a result, transient faults (soft errors) tend to appear more frequently than in previous manufacturing generations, which, in turn, makes them a major threat to reliable computing system operation. However, not all faults will necessarily harm the system stability nor will have the same severity for software workloads. Several metrics have been proposed to express the system vulnerability against transient faults. The Architectural Vulnerability Factor (AVF) [17] was proposed by Mukherjee *et al.* to quantify a microprocessor’s vulnerability to transient faults; the AVF of a microprocessor hardware structure is the probability that a fault in it will affect the correct execution of a program. Similarly, IVF [18] and H-AVF [19] have been proposed to express the vulnerability of microprocessor structures to intermittent and permanent (hard) faults, respectively. Sridharan and Kaeli later proposed finer levels of abstraction for the AVF, introducing the Hardware Vulnerability Factor (HVF) [20] and the Program Vulnerability Factor (PVF) [21], each to quantify the portion of AVF that can be attributed to the Hardware layer (including the microarchitecture) and the Software layer, respectively.

The assessment of microprocessor vulnerability to soft errors is very crucial during the early design stages to guide efficient error protection. Fault-tolerance mechanisms impose area, power and performance overheads, which can, for example, vary in a range between 1% and 125% (in terms of extra memory capacity) for typical memory error detection and correction [17]. Thus, significant effort must be devoted to effectively measure the vulnerability of a CPU design as early as possible and make cost effective design decisions for error protection.

Multi-bit transient upsets (MBU) are an increasingly important challenge in SRAMs [1], [8]. High-energy particles are able to deposit charge onto multiple SRAM storage cell because of their shrinking dimensions, and this results in upsets in multiple adjacent bits. This is called a spatial multi-bit fault (sMBF) (or multi-bit upset (MBU)). Ibe *et al.* in [1] show that in a 180nm fabrication technology node, about 5.3% of neutron-induced errors in SRAMs are multi-bit upsets, while in 22nm process, this ratio is as high as 45%. This increase in multi-bit fault rate is projected to continue despite the introduction of technologies such as FinFET transistors, which have reduced fault rates [22] [23]. Estimating architectural vulnerability factors of microprocessors hardware structures by considering multi-bit faults at their rates for a particular technology, allows architects to have

This work has been funded by the European Union through the CLERECO project (FP7 Grant 611404), the UniServer project (H2020 Grant 688540), a Tetramax-Bilateral-TTX-2 grant, and by Cisco Research.

a more accurate evaluation of the system’s vulnerability and protect the structures for realistic upsets rates and cardinalities and not only for single-bit faults. Design-time vulnerability estimation techniques (including fault injection and lifetime analytical methods) focus on single-bit faults and fail to capture the effects of multi-bit faults [3] [6] [28] [33] [34] [35] [36].

In this paper, we perform a microarchitecture-level reliability assessment using fault injection at the full system level. We present a comprehensive study and analysis that considers reportedly important spatial multi-bit faults, in 6 important microarchitectural components of an ARM Cortex-A9 CPU, in order to quantify the effect of multi-bit upsets in the overall system reliability. We employ and extend GeFIN [3] [6], a Gem5-based [2] fault injection framework to inject multi-bit faults in the hardware structures of the CPU model. We enhance the fault mask generator of GeFIN to support the generation of spatial multi-bit faults in adjacent areas of a structure. The results of this study reveal that, for important hardware structures such as the L1I cache, the vulnerability of triple-bit faults increases up to 3.2x (220%) when compared to single-bit fault injection, for 22nm fabrication technology; a huge gap in soft error vulnerability assessment that our analysis reveals. We also show that the portion of the overall CPU FIT rate that is caused by multi-bit faults is significant in all technology nodes after 250nm, it consistently increases and reaches 21% at 22nm. Even larger contributions of multi-bit faults in the CPU FIT rates in latest technologies (14nm, 10nm, 7nm) can be safely assumed, calling for diligent protection against multi-bit upsets at least in the hardware structures that our study identifies as the most vulnerable. The presented vulnerability estimation methodology can be used in all current and future technologies as the supported fault models are very flexible and easily configurable.

II. RELATED WORK

The majority of early vulnerability estimation studies focus on single bit faults and only a few consider the impact of multi-bit faults in SRAMs and logic [8] [23] [24] [25] [26] [27]. George *et al.* in [8] modeled only spatial double-bit faults using microarchitecture level fault injection on PTLsim simulator and reported the AVF of only two hardware components (the register file and reorder buffer), while the multi-bit rates across technologies have not been taken into consideration to quantify the contribution of multi-bit faults in the overall AVF and FIT measurement. In [23], the authors propose the use of ACE analysis to capture the vulnerability of multi-bit faults, assuming the existence of hardware protection. They also use GPU fault injection to validate the assumption that bit ACEness (importance for architecturally correct execution) is not affected by neighboring bits in a structure. In this work, we do not make assumptions of hardware protection as we consider that this is the scope of performing a reliability evaluation in early design phases – based on the findings of our analysis informed multi-bit error protection can be implemented in a CPU design.

The authors in [24] [25] [26] [27] perform multi-bit reliability analysis at the software or architecture level (not the actual hardware or microarchitecture level that we focus on). Lu *et al.* in [24] compare the results of single-bit faults to double-bit faults in a single word and in different words at the LLVM compiler’s intermediate code level. They find that there is not much variation between the error resilience of the different models.

The main focus of the work is on the fault injection tool rather than a thorough study of the impact of multiple bit-flip errors. Ayatollahi *et al.* in [25] compare the single bit-flip model with the double bit-flip model at the assembly-level code. In their study, double bit-flip errors are only injected into a single word (i.e., register or memory location). They also find that the SDC results obtained for the two fault models are only marginally different. Adamu-Fika and Jhumka in [26] compare the results of injecting double-bit faults in a single word to those into different words at the LLVM compiler’s intermediate code level. Similar to the previous studies, the results of their experiments show that, on average, the difference between the percentages of data failures for the two models is marginal. On the contrary, Sangchoolie *et al.* in [27] presented that the single-bit fault experiments can give results close to multi-bit fault experiments, while they propose that at most triple-bit faults are reasonable.

At the manufacturing technology level multi-bit fault studies have clearly revealed the importance of accounting and designing for multi-bit faults in modern hardware. The authors of [1] [41] [42] [43] [44] [45] analyze the ways that multi-bit upsets affect SRAM cells, while in [39] [46], the authors examine protection options for spatial multi-bit faults. Suh *et al.* [40] introduce a periodic autoregressive moving average (PARMA) framework to compute the mean time to failure (MTTF) of caches from single-bit faults and multi-bit upsets. In [47], the authors consider several design characteristics and apply soft-error rates in different technology nodes to showcase how the error rates affect the same design on different technologies.

Microarchitecture level fault injection has been used in various studies [3] [6] [28] for assessing reliability on hardware components basis, but also for capturing the performance deviation caused by the presence of faults in speculative components [29] [30] [31] [32]. There are also studies [33] [34] [35] [36] [37] [38] that spread their focus beyond a single abstraction layer and aim to exploit the benefits of multiple abstraction layers to either accelerate the evaluation process or deliver cross-layer reliability evaluation.

In this work, we move beyond previous studies and present a complete analysis of *realistic multi-bit* failure rates which aggregates the effects of multi-bit upsets (single, double, triple) for 8 different technologies, 6 major microprocessor components and 15 different workloads. We quantify in fine granularity the AVF and FIT rates of all the above combinations (technology, component, workload) in an out-of-order ARM CPU model and demonstrate the significant gap between single-bit analysis and realistic multi-bit vulnerability analysis which corresponds to the actual failure modes that a real CPU experiences in the field.

III. EXPERIMENTAL SETUP

Using a very extensive fault injection, simulation-based study, we collected a set of interesting observations that demonstrate the accuracy improvement we contribute to the vulnerability estimation process of modern microprocessors. Our benchmarks base consists of 15 benchmarks, simulated for 6 hardware components in an ARM Cortex-A9 with one, two and three faults injected per run for each component. This corresponds to 540,000 injections for all components and benchmarks used in this study (details for the statistical properties of the fault injection campaigns follow in the next subsections).

TABLE I. SUMMARY OF SETUP ATTRIBUTES.

Microarchitectural attribute	Value
ISA / Core	ARMv7 / Out-of-Order
L1 Data cache	32KB 4-way
Clock Frequency	2 GHz
L1 Instruction cache	32KB 4-way
L2 cache	512KB 8-way
Data / Instruction TLB	32 entries
Physical Register File	56 registers
Instruction queue	32
Reorder buffer	40
Fetch / Execute / Writeback width	2 / 4 / 4

A. Platform

Among the available abstraction models, microarchitecture level is the only one that comes with sufficient hardware detail and is capable of running full-system simulation including the operating system. While RTL is closer to the hardware, its extremely small simulation throughput does not allow running of multiple fault injection simulations of a full system stack that includes an operating system and transactions with I/O peripheral devices [48]. Our microarchitectural modeling is based on Gem5 simulator, the state-of-the-art, flexible full system cycle-accurate simulator [2]. Gem5 fully supports ARM ISA and comes along with a detailed out-of-order core implementation. The CPU core was configured to resemble the microarchitecture of ARM[®] Cortex[™]-A9. Table I presents the configuration of the Gem5 CPU core.

GeFIN fault injection framework [3] was used on top of Gem5 for the reliability assessment. We have extended the GeFIN infrastructure with a new fault generator to inject spatial multi-bit upsets of different geometries and cardinalities during the simulation of the system. The faults were injected in 6 important components that correspond to more than 94% of the memory cells of the CPU: L2 Cache, L1 Data and Instruction Caches, Physical Register file, Data and Instruction Translation Lookaside Buffers (TLB). For each of the 6 components, three sets of 2,000 single-bit faults, 2,000 double-bit faults, and 2,000 triple-bit faults were generated, resulting in 36,000 fault injection simulations per workload (or 540,000 fault injections for all 15 benchmarks together). We follow the widely adopted formulation of [4] for the statistical fault sampling calculations; our 2,000 faults samples choice corresponds to 2.88% error margin with 99% confidence level. This estimation corresponds to an initial unknown AVF estimation [5], which, as suggested by [4], is set to $p = 0.5$ in order to maximize the fault sample. After the execution of simulation campaign, however, we can re-adjust the p variable in the formula with the result of the estimation, shifted by the maximum error margin. This gives us a better estimation of the actual error margin for each combination of workload/component, which in our results is between 2.88% and 2.4% with 99% confidence.

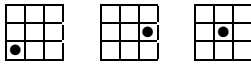
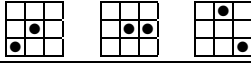
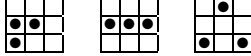
Microarchitecture level fault injection offers high observability, allowing distinction of where exactly the faults strike (e.g., whether it was on kernel or user mode or data, whether the corrupted entry was used or not, etc.) but also detailed information of what was the system effect as we describe in the following subsection.

B. Fault Modeling

Spatial multi-bit faults can affect both neighboring cells that share a p-well or an n-well as well as memory cells in a nearby area. Based on the findings of [1] and [43], we have created a new fault generator for GeFIN that implements the concept of a fault cluster [1]. For a given cluster size of X rows and Y columns, the generator creates N random faults (bit flips) inside this array. The cluster is then placed in a random position inside the SRAM array, where it marks where the faults should be injected. Cluster sizes and fault numbers can be configured to resemble observed patterns of fabrication technologies. The vulnerability of each cluster is quantified separately and its contribution to the overall AVF of a device is weighted according to the technology characteristics. This level of configuration allows assessment of current and future technology nodes.

Our experiments consist of single-bit, double-bit and triple-bit faults generated in a 3x3 cluster (quadruple-bit and larger rates are virtually zero as reported in [1]). A generated fault mask contains a multi-bit fault that is injected to the hardware of the CPU. This means that for each experiment we virtually isolate a 3x3 cluster of a structure. Then, we randomly select which are the faulty cells in this cluster and perform the injection. We know that those bits are placed in adjacent areas, very similarly to what is performed during accelerates beam testing where the multi-bit faults are observed in adjacent areas of memory blocks. Table II shows some examples of single-bit, double-bit and triple-bit upsets in a 3x3 cluster produced by our faults generator. Notice that, unlike MBU code that is used in [1], which defines as cluster size the smallest possible region that can fit all faults, for the fault generation we can also have series of faults that can fit in smaller clusters (e.g., the first example of double-bit could fit in a 2x2 cluster). This way we model more realistic results with all smaller sub-clusters included in our analysis.

TABLE II. EXAMPLES OF MULTI-BIT UPSET CATEGORIES IN A 3 X 3 CLUSTER USED IN OUR ANALYSIS.

Category	Error Bit Pattern Example
Single-bit Fault	
Double-bit Fault	
Triple-bit Fault	

C. Fault Effect Classification

The GeFIN injector classifies the outcomes of each fault simulation based on the impact of the fault on the simulated system. Five classes are used for the fault effects classification for AVF measurements:

Masked: Masked includes the fault injection runs in which the fault does not affect the execution of the application (which is executed through its end) or the system. The result of a simulation with a masked fault is identical to the fault-free simulation in terms of the output of the application and any exceptions generated during execution.

Silent Data Corruption (SDC): Silent Data Corruption includes the fault injection runs for which the final output of the program that is written to an output file is corrupted (differs from the output of the fault-free execution) and no other indication of the fault has been recorded (an abnormal event such as an exception, etc.).

Crash: Crash includes any case that results in an unrecoverable situation that stops the simulated program. Crashes involve: a process crash, where the simulated program was abnormally terminated and a system crash (kernel panic), where the simulated full-system was unable to recover.

Timeout: Timeout includes all the cases where the simulation did not finish within a certain amount of time, that lead to either a Deadlock (a condition in which the program flow has been trapped -due to the injected fault- and can't commit any further instructions) or a Livelock (a situation where the program flow has been redirected -due to the injected fault- and continues the execution of instructions on random code areas). The execution timeout limit to monitor these cases is four times equal to the fault-free execution of each benchmark.

Assert: Assert includes all the cases where the simulation was unexpectedly terminated due to a simulator failure. If the

simulator crashes or reaches a high-level condition that is unable to handle, it raises an assertion to stop the simulation.

D. Benchmarks

In our experiments we use 15 workloads from the MiBench benchmarks suite [7], as shown in Table III. The suite is commonly used in reliability studies [3] [6] [8] [9] [10] [11], as it combines benchmarks with reasonable execution (thus simulation) time and facilitates complete end-to-end executions for the thousands of fault injections required in our comprehensive analysis. The suite also includes programs from diverse application domains that share data and control flow characteristics with SPEC benchmark suite [12]. AVF is estimated using the program output and thus, complete execution of the workloads is required.

To have a broad analysis and avoid the bias of results on specific applications, we have chosen benchmarks with different computational characteristics. Table III lists the benchmarks and shows the execution time for each benchmark.

IV. FINE-GRAIN EXPERIMENTAL RESULTS

In this section, we present the detailed results from our experiments. The results report the vulnerability (AVF) for different microprocessor components (L1D, L1I, L2 caches, Register File, DTLB and ITLB) of the analyzed out-of-order CPU (ARM Cortex-A9) for 8 technology nodes, and the FIT of each component and technology node. The AVF of different components scales differently from single-bit to multi-bit faults and we elaborate on these differences.

A. L1 Data (L1D) Cache

As we can see in the single-bit fault bars of Fig. 1, the masked faults vary between 53.7% (cjpeg) and 94.5% (stringsearch), and the SDC vulnerability between 39.4% (cjpeg) and 1.1% (dijkstra). SDC is the dominant fault effect class, while the crashes are estimated between 2-3% for most of the cases. The vulnerability of the remaining two classes (Assert and Timeout) is extremely low, with the only exception of qsort, in which the Timeouts account for 5.5% of the total.

In the double-bit fault bars of Fig. 1, the masked faults vary between 39.2% (rijndael_dec) and 92.1% (stringsearch). We observe that, in double-bit fault experiments, all the benchmarks

TABLE III. BENCHMARK EXECUTION TIME.

Benchmark	Execution Time (clock cycles)
CRC32	132,195,721
FFT	48,339,852
ADPCM decode	53,690,367
basicmath	67,556,250
jpeg C	26,126,843
dijkstra	41,643,556
jpeg D	10,105,853
gsm_dec	12,862,888
qsort	31,326,716
rijndael D	33,327,494
sha	12,141,593
stringSearch	1,082,451
susan_c	2,150,961
usan_e	2,876,202
susan_s	13,750,557

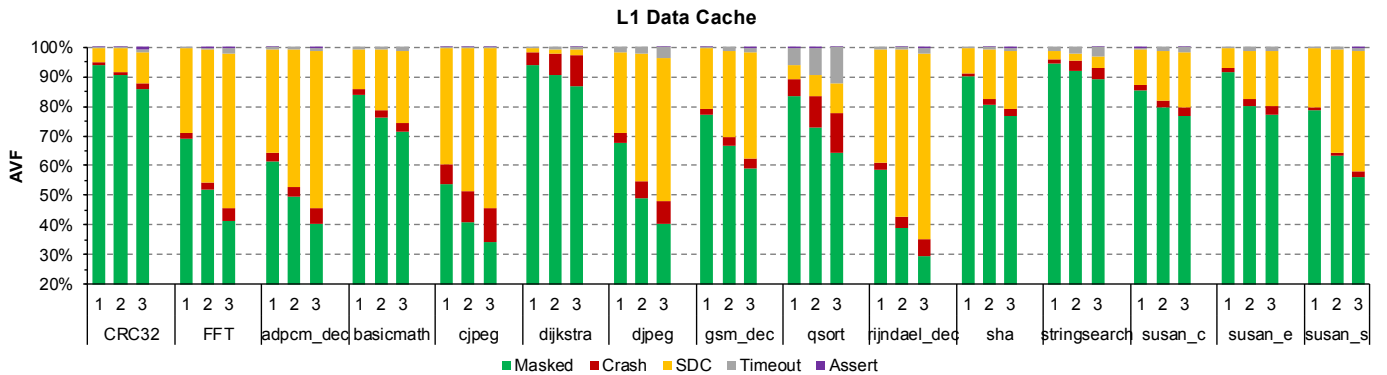


Fig. 1. AVF for single-bit, double-bit and triple-bit fault injection campaigns for 15 benchmarks for L1 Data Cache (L1D Cache).

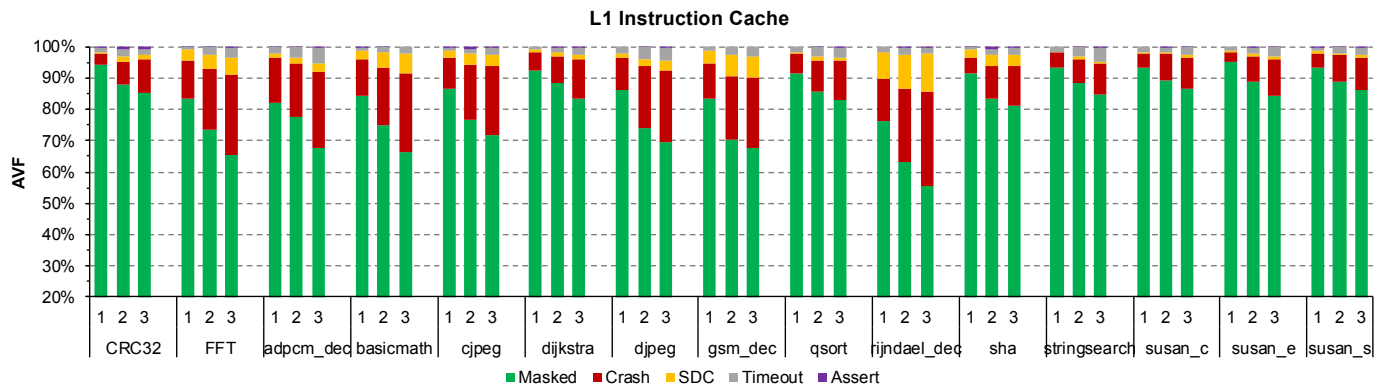


Fig. 2. AVF for single-bit, double-bit and triple-bit fault injection campaigns for 15 benchmarks for L1 Instruction Cache (L1I Cache).

report higher vulnerability compared to the single-bit fault injection. SDC is estimated between 56.3% (rijndael_dec) and 1% (dijkstra). Crash vulnerability varies between 10.6% (qsort) and 1.3% (susan_s), however, only one benchmark (cjpeg) has a crash rate higher than 10% (10.5%). Although the vulnerability increases in double-bit faults, the balance between fault-effect classes is not changed.

The highest vulnerability is reported in triple-bit fault bars of Fig. 1, where the masked faults vary between 29.7% (rijndael_dec) and 88.9% (stringsearch). Similarly to the double-bit faults, SDCs dominate among the vulnerable classes, which varies between 62.5% (rijndael_dec) and 1.5% (dijkstra). Crash vulnerability varies between 13.4% (qsort) and 1.8% (CRC32). For the two remaining classes (Assert and Timeout) the vulnerability is extremely low, except for the case of qsort, in which there is a 12.3% of timeouts, which is higher than SDC (10%), and slightly lower than crash (13.4%).

B. L1 Instruction (L1I) Cache

Fig. 2 presents the AVF results of L1 instruction cache for single, double and triple-faults. In the single-fault results, the masked faults vary between 76.3% (rijndael_dec) and 95.1% (susan_e). Crash is the most prominent faulty behavior among the vulnerable cases, as faults in L1I Cache affect the instructions and immediate values, while the SDC vulnerability is extremely low for most of the benchmarks. Specifically, the crash vulnerability varies between 14.3% (adpcm_dec) and 3.3% (susan_e), and 6 out of 15 benchmarks provide a rate higher than 10%. The SDC vulnerability varies between 8.6% (rijndael_dec) and 0.1% (stringsearch). Similar to L1D cache, the fault rates for the rest of the classes are extremely low (i.e., Timeouts) or virtually zero for the Assert class.

As we can see in the double-bit fault bars of Fig. 2, the masked faults vary between 63% (rijndael_dec) and 89.3% (susan_c). The vulnerability is significantly lower compared to L1D Cache, where we can benchmarks with AVF more than 50%. Similarly, double-bit AVF is always higher compared to single-bit faults. The crash vulnerability varies between 23.4% (rijndael_dec) and 7.2% (CRC32), and SDCs vulnerability is lower than 10% in all cases except for rijndael_dec, which provides an SDC vulnerability of 11.1%. Assertions are extremely low (lower than 1% in all cases), while Timeouts are lower than 4% for all benchmarks.

In the triple-bit fault bars of Fig. 2, the masked faults vary between 55.4% (rijndael_dec) and 86.7% (susan_c). Interestingly, the measured vulnerability is never more than 50%, even for triple-bit faults. The crash vulnerability varies between 30.2% (rijndael_dec) and 9.7% (susan_c). Compared to the single-bit fault injections crashes escalate to almost 2x for most benchmarks, while for the cases where the vulnerability is low, we observe a 3x increase in crash vulnerability (susan_e from 3.3% to 11.6%). SDC cases vary between 12.1% (rijndael_dec) and 0.5% (stringsearch), with all the cases, except for rijndael_dec, to be under 10%. We can also see a timeout ratio of 4.8% for adpcm_dec. Assertions in the instruction cache are marginal.

C. L2 Cache

L2 cache vulnerability is presented in Fig. 3. Single-bit AVF varies between 47.8% (adpcm_dec) and 4.2% (stringsearch), sharing similarities with L1 Data cache. SDC and Crashes are the dominant fault effect classes. SDC vulnerability varies between 44.4% (adpcm_dec) and 0.1% (sha), while crashes vulnerability varies between 4.7% (susan_s) and 1.9% (rijndael_dec). Timeout and assertions are negligible.

When considering double-bit fault bars of Fig. 3, the masked faults vary between 40% (adpcm_dec) and 91.7% (sha), showing a difference of over 50% between these two corner cases. adpcm_dec is the only benchmark with a vulnerability greater than 50%. L1D Cache had 4 benchmarks in the same category, while L1I has none. SDC vulnerability varies between 54.4% (adpcm_dec) and 0% (susan_s). This actually shows a reduction in SDCs for susan_s, from 0.2% in single-bit faults to 0% in double-bit faults. Crash vulnerability varies between 9.6% (susan_s) and 3.3% (qsort). As a result, given that susan_s provides zero SDC vulnerability, this leads to a 2x increase of crash vulnerability (from 4.7% to 9.6%). Timeouts vary between 2% (qsort and susan_c) and 0.75% (cjpeg).

As we can see in the triple-bit fault bars of Fig. 3, the masked faults vary between 30.2% (adpcm_dec) and 89.8% (stringsearch). Compared to single and double-bit fault injection campaigns, for the L1D Cache and the L1I Cache, we can notice that the L2 cache has similar behavior to the L1D Cache, making the L1I Cache less vulnerable. In L1I Cache, we notice only 2 benchmarks out of 15, with a rate lower than 50% for the masked faults, which is far better than L1D Cache's rate, in which 5

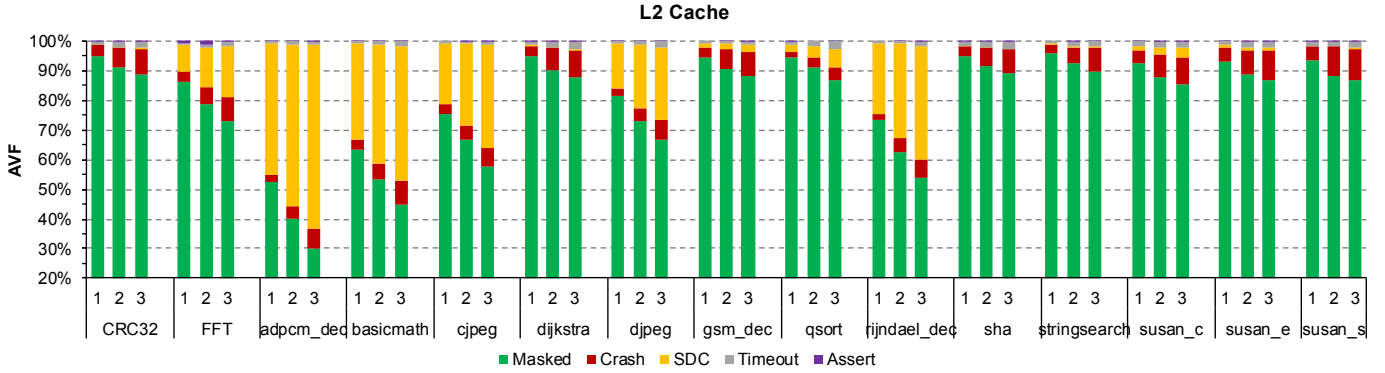


Fig. 3. AVF for single-bit, double-bit and triple-bit fault injection campaigns for 15 benchmarks for L2 Cache.

benchmarks has masked faults lower than 50%. SDC vulnerability varies between 62.1% (adpcm_dec) and 0.2% (susan_s), while crash vulnerability varies between 10.9% (susan_s) and 4.4% (qsort). Timeouts vary between 2.7% (qsort) and 1% (adpcm_dec), while assertions continue holding negligible rates, with a maximum of 0.5% for FFT; lower than single- and double-bit fault injection campaigns.

D. Register File

The register file is the only pipeline structure in our study. Fig. 4 shows that the masked single-bit faults vary between 84.2% (sha) and 94.4% (susan_c). The register file is less vulnerable compared to all other components. Similarly to the cache memories, SDCs and Crashes dominate among the other vulnerable classes. More specifically, crashes vary between 9.3% (susan_s) and 3.1% (stringsearch), and SDCs between 11% (sha) and 0.5% (susan_c). Timeouts and assertions have very low rates, with timeouts reaching as large as 2.4% (stringsearch) and assertions have 0% for 13 out of 15 benchmarks.

Double-bit fault bars show that the masked faults vary between 74% (sha) and 89.7% (stringsearch). In 13 out of 15 benchmarks, crashes are the most prominent faulty behavior among the other vulnerable cases, which varies between 14.9% (susan_s) and 5% (stringsearch). SDC vulnerability varies between 15.5% (sha) and 0.7% (stringsearch). Timeouts vary between 4.5% (stringsearch) and 1.3% (FFT).

As we can see in the triple-bit fault bars of Fig. 4, the masked faults vary between 69.1% (sha) and 85.5% (stringsearch). The AVF difference among single, double and triple-bit faults is the smallest, compared to the other components. The highest reported vulnerability (69.1%) is also the lowest among the six components for multi-bit faults. The second most frequent fault effect is a Crash. Crashes vary between 18.4% (susan_s) and 7.7% (stringsearch), with 14 out of 15 benchmarks having ratios higher than 10%. SDCs vary between 18.5% (sha) and 0.8% (susan_c). Timeouts vary between 5.7% (stringsearch) and 2.1% (gsm_dec), and assertions reach a high of 0.3% for stringsearch.

E. Data Translation Look-Aside Buffer (DTLB)

TLBs do not directly participate in data transactions of a program, but instead, they facilitate address translation. Faults in TLBs can primarily lead to multiple incorrect memory accesses. Both TLBs are highly vulnerable to all of the evaluated fault models. As we can see in the single-bit fault bars of Fig. 5, the AVF varies between 57.6% (basicmath) and 39.2% (qsort). We notice that 7 out of 15 benchmarks have AVF higher than 50%. The most frequent failure sourcing in DTLB faults appears to be the Crashes followed by Timeouts. Crashes vary between 32.8% (FFT) and 16.7% (qsort). Interestingly, we observe high rates for timeouts which vary between 23.1% (stringsearch) and 6.3% (cjpeg), while SDCs have lower rates for most of the cases, which range between 17.1% (basicmath) and 1.5% (sha). For the DTLB we also observe higher ratios for assertions compared to

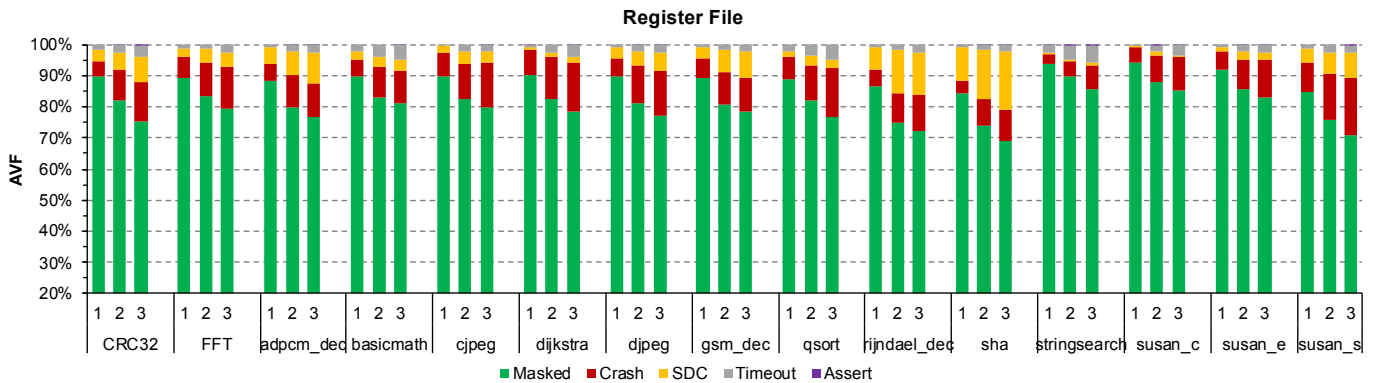


Fig. 4. AVF for single-bit, double-bit and triple-bit fault injection campaigns for 15 benchmarks for the Register File.

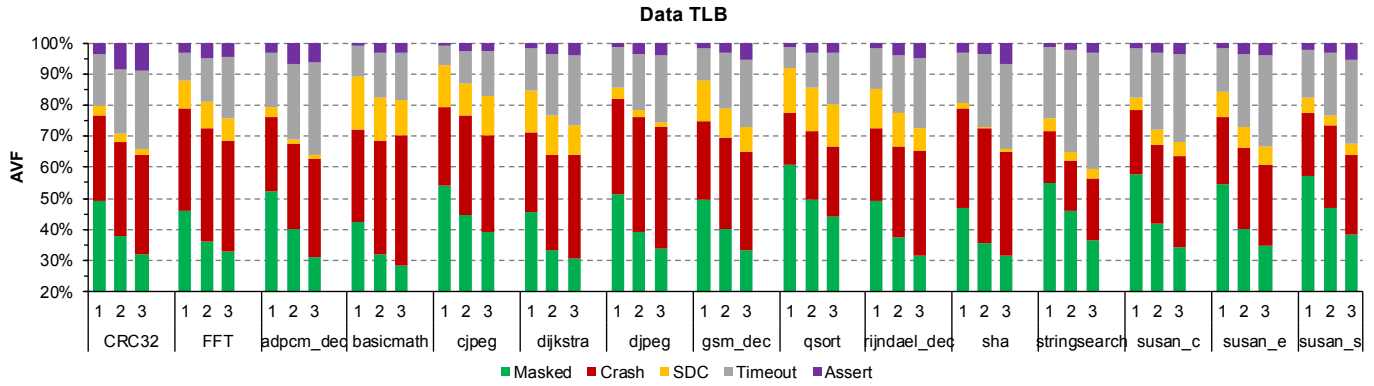


Fig. 5. AVF for single-bit, double-bit and triple-bit fault injection campaigns for 15 benchmarks for Data TLB (DTLB).

the previous components, with a maximum value of 3.50% for CRC32.

For double-bit faults, the vulnerability is between 67.9% (basimath) and 50.3% (qsort). In all cases, AVF is higher than 50%. We observe a vulnerability escalation similar to the other components. Crashes vary between 37.2% (djpeg) and 16.2% (stringsearch). Timeouts vary between 32.8% (stringsearch) and 10.2% (cjpeg). SDCs vary between 14.1% (basimath and qsort) and 0.6% (sha). Interestingly, in most benchmarks (14 out of 15) the highest SDC rates are reported in single-bit faults. Assertions vary between 8.6% (CRC32) and 2.3% (stringsearch), which is approximately 2x the single-bit campaign for most of the cases.

As we can see in the triple-bit fault bars of Fig. 5, the vulnerability increases, from 55.9% (qsort) to 71.5% (basimath). Timeouts vary between 37.5% (stringsearch) and 14.8% (cjpeg). SDCs vary between 13.4% (qsort) and 0.7% (sha). As the number of faults increases, we observe a reduction on SDC rates (similar to double-bit faults), in which 11 out of 15 benchmarks have lower rates compared to the double-bit fault injection campaigns. DTLB reports the worst rates for assertions compared to other components since faults in the DTLB can cause a physical address request that is not part of the system map. We can see a variation from 8.9% (CRC32) to 2.5% (cjpeg). Assertions are more frequent than SDCs for 6 out of 15 benchmarks.

F. Instruction Translation Look-Aside Buffer (ITLB)

Like the DTLB, ITLB guides the translation of the instruction fetch requests. Faults can result to incorrect physical address

translation that are used in the program counter. Single-bit fault bars of Fig. 6 show that AVF varies between 41.9% (rijndael_dec) and 62% (basimath). We observe that similarly to the DTLB, 6 out of 15 benchmarks the AVF is more than 50%. Crashes are the most frequent fault effect, estimated between 43.1% (djpeg) and 1.5% (stringsearch). On the other hand, timeouts are estimated between 16.2% (qsort) and 44.2% (stringsearch). Interestingly, while the stringsearch benchmark has the lowest ratio of crashes, it has the highest ratio of timeouts. Faults in ITLB are expected to lead to either a crash of the benchmark because of a virtual address in the TLB that does not correspond to a mapped physical address or to a Timeout if the program counter is guided to a random set of instructions. Thus, it is unlikely to have an SDC. Most of the benchmarks (10 out of 15) have zero SDCs, while the highest rate is just 0.4% (8 out of 2000 runs led to an SDC) for sha benchmark. Assertions also appear in small rates, with a maximum of 1.3% for CRC32.

Similarly to the DTLB, all the benchmarks have a vulnerability greater than 50% in the double-bit fault bars of Fig. 6, the AVF varies between 54% (rijndael_dec) and 69.8% (basimath). Crashes are estimated from 1.5% (stringsearch) to 44.3% (djpeg), while timeouts vary between 55.5% (stringsearch) and 22.1% (djpeg). SDCs, still hold negligible rates while assertions reach a high of 2.1% for CRC32.

ITLB is the most vulnerable among all other components. Crash vulnerability varies between 46.4% (basimath) and 1.4% (stringsearch) for triple-bit faults, while timeouts vary between

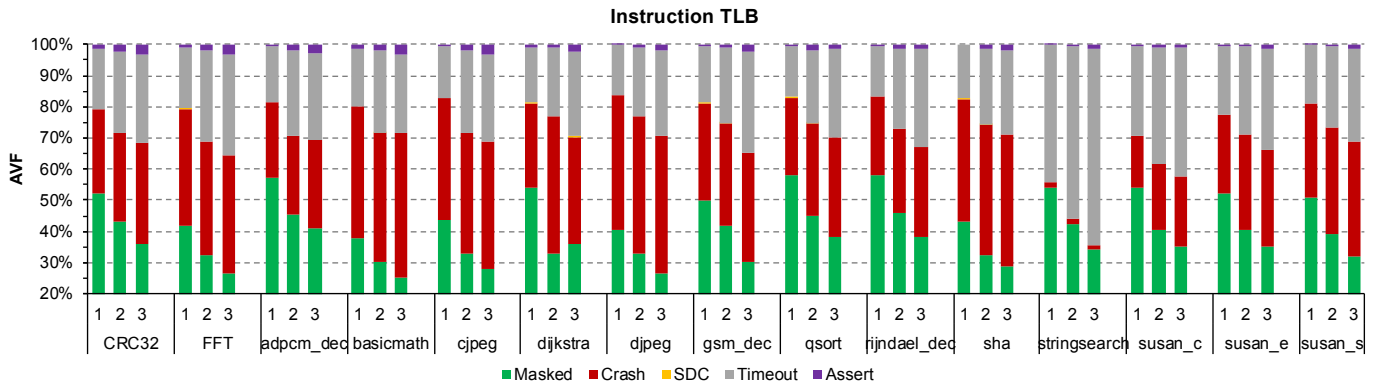


Fig. 6. AVF for single-bit, double-bit and triple-bit fault injection campaigns for 15 benchmarks for Instruction TLB (ITLB).

62.8% (stringsearch) and 25.4% (basimath). Interestingly, with triple-bit fault campaigns, we did not observe any SDCs apart from a single exception in all in 30,000 runs, showing that faults in ITLBs cannot really result in SDCs. Assertions appear to have the highest ratio of 3.4% for CRC32 (almost 3x higher than the single-bit fault campaign).

G. Summary and Observations

Table IV summarizes the observations of the previous subsections regarding the AVF estimation, for each component across all benchmarks between single- and double-bit faults (2-bit column) and between single- and triple-bit faults (3-bit column). Table IV clearly shows that the L1I cache suffers the most from triple-bit faults, as the vulnerability is 3.2x higher (220% increase) compared to single-bit faults, while the most sensitive component to double-bit faults is the L1D cache, which is 2.4x more vulnerable (140% increase) than in single-bit faults. On the contrary, the DTLB is the most resistant in double-bit fault injections among all other components (1.4x increase), and the ITLB shows the smallest effect of triple-bit faults (1.5x increase). Another important observation is that the TLBs show a different faulty behavior than other components, in which we observe a lower SDC rate (because faults in TLBs primarily lead to incorrect memory accesses, and thus, wrong fetched data or instructions from memory), and the timeouts and assertions present significantly higher vulnerability than other components.

TABLE IV. VULNERABILITY ANALYSIS CONCLUSIONS AND VULNERABILITY DIFFERENCE PER COMPONENT.

	Comments	Vulnerability Increase	
		2-bit	3-bit
L1D Cache	<ul style="list-style-type: none"> All benchmarks have significantly lower rate of masked faults compared to single-bit fault injections SDCs have the highest rate among the other vulnerable cases Timeouts and assertions are low 	2.4x	2.7x
L1I Cache	<ul style="list-style-type: none"> All benchmarks have masked fault rate greater than 50% SDCs have the highest rate among the other vulnerable cases Timeouts and assertions are low 	2.3x	3.2x
L2 Cache	<ul style="list-style-type: none"> SDC and crash vulnerability have the highest rates It is more vulnerable to multi-bit faults than L1D and L1I caches Timeouts and assertions are low 	1.9x	2.4x
Register File	<ul style="list-style-type: none"> SDCs and Crashes dominate among the other vulnerable cases It has the smallest (absolute) vulnerability increase among all components, reporting the highest level of multi-bit fault tolerance. Timeout and assertions are low 	2.1x	2.7x
DTLB	<ul style="list-style-type: none"> All the benchmarks have masked fault rate lower than 50% It appears to have the worst effect to multi-bit faults, even in single-bit faults Timeouts and Crashes are the most dominant faulty behaviors 	1.4x	1.6x
ITLB	<ul style="list-style-type: none"> Similar to DTLB, but ITLB has no SDC It is relevant to instructions; we observe virtually zero SDC vulnerability 	1.5x	1.5x

V. AVF PER TECHNOLOGY NODE – ALL MULTI-BIT FAULTS

In the previous section, we discussed in details the increase of the vulnerability to individual classes of spatial multi-bit faults (double, triple) in all six components of our analysis. In this section, we quantify the aggregate effect of these vulnerabilities to the total AVF of each component. Every fabrication technology node suffers different rates of multi-bit upsets and patterns. Denser technologies tend to suffer from MBUs in larger cluster sizes and, depending on the fabrication technology, potentially higher rates. FinFET, for instance are reported to be less prone to faults compared to CMOS [22], [23]. These attributes are inherited from the technological layer and applied to our methodology, which accordingly applies the contribution of each fault-model to the device’s reliability estimation. We utilize the per-component and per multi-bit class vulnerabilities of the previous section, to calculate the total AVF for every technology node and every hardware structure.

A. AVF per Hardware Structure

To comprehensively summarize the detailed data and results of the previous section, we average the AVF of each component across the 15 different benchmarks. However, instead of calculating the straightforward arithmetic mean of the AVFs of the component for the different benchmarks, we weighted the AVFs according to the execution time of the benchmarks. Thus, very short benchmarks will have a smaller impact on the component’s AVF compared to longer ones. The resulting weighted average AVF takes into consideration the execution time (measured in clock cycles) of each of the 15 benchmarks. It is calculated by summing the AVF of all benchmarks, each multiplied by the execution time of the corresponding benchmark and divided them by the sum of the execution time of all the benchmarks, as shown in equation (2):

$$W_{AVF}(c) = \frac{\sum_{k=1}^N AVF_k(c) \cdot t_k}{\sum_{k=1}^N t_k} \quad (2)$$

where, $W_{AVF}(c)$ is the weighted AVF of a c component per fault number, $AVF_k(c)$ is the AVF of a c component per benchmark, t_k is the execution time of each benchmark, and N is the total number of benchmarks. The AVF numbers in this formula can be either the single-bit AVFs alone, the double-bit AVFs, alone, the triple-bit AVFs alone, or finally the aggregate AVFs including all cardinalities weighted by the technology rates.

Table V presents the technology-independent weighted average AVF for the single, double and triple-bit faults separately for each component. The observations of the previous sections are also demonstrated in Table V (e.g. that the TLBs are the most vulnerable components). The vulnerability is increased as we increase the number of injected faults (from single to double and from double to triple). The TLBs have higher AVF rates than the other components, even for single fault injection. The probability of a fault in the TLB creating an error is always larger than 50%. The Register File has the smallest AVF rates among all components, while in the case of the cache memories, we can see that the L1D Cache has the highest AVF rates among all cache levels. As far as the percentage increases are concerned, we observe that for every component there is an increase in the

TABLE V. WEIGHTED AVF PER COMPONENT FOR 1, 2, AND 3 FAULTS.

Component	Injected Faults	AVF	Percentage Increase
L1 D Cache	1	20.32%	-
	2	29.70%	+46.16%
	3	36.28%	+22.15%
L1 I Cache	1	12.01%	-
	2	19.57%	+62.95%
	3	25.14%	+28.46%
L2 Cache	1	17.94%	-
	2	24.83%	+38.4%
	3	30.13%	+21.35%
Register File	1	10.95%	-
	2	18.65%	+70.32%
	3	23.01%	+23.38%
ITLB	1	50.31%	-
	2	62.91%	+25.04%
	3	66.67%	+5.98%
DTLB	1	50.66%	-
	2	61.77%	+21.93%
	3	67.22%	+8.82%

AVF as the number of faults increases, but most importantly we notice that the increase between single- and double-bit faults is larger for all components compared to triple-bit faults. For example, when double-bit faults are injected in Register File, we observe an AVF increase, which is as high as 70.32%, compared to single-bit fault injection, and it is the larger increase among all components. On the other hand, the lowest AVF increase among all components for triple-bit fault injections compared to double-bit faults injections is observed in ITLB, which is 5.98%.

These numbers correspond to the vulnerability of Gem5 configuration of the ARM[®] Cortex[™]-A9 microarchitecture, and are independent of the manufacturing process.

B. Aggregate Multi-bit AVFs per Technology Node

Every particle-induced soft-error has a different probability to result in a multi-bit upset at different fabrication technologies. These are shown in Table VI. As the multi-bit upset rates for 4-bit faults and above are very low, we add them to the triple-fault class. We use the multi-bit upset ratios from 250nm to 22nm technology nodes, as presented in [1]. For more consistency, we used a single source for the technological data and thus, we excluded more recent technologies, such as FinFET 14 nm and 7 nm; our estimation is fully applicable to those nodes as well.

TABLE VI. MULTI-BIT RATES PER NODE.

Technology Node	Single-bit Faults	Double-bit Faults	Triple-bit Faults
250nm	100.00%	0.00%	0.00%
180nm	96.40%	3.60%	0.00%
130nm	93.40%	4.40%	2.20%
90nm	87.80%	9.60%	2.60%
65nm	81.60%	16.10%	2.30%
45nm	72.20%	23.00%	4.80%
32nm	65.30%	29.10%	5.60%
22nm	55.30%	34.40%	10.30%

As Table VI shows, in older fabrication technologies, the probability of appearance of a multi-bit fault are very small. On the other hand, as the transistor sizes shrink, the probability of multi-bit faults becomes significantly higher. By combining the probabilities for each fault class (single, double, triple) with the corresponding AVF for each component, we calculate the AVF of the component for each technology node. We use the following formula per technology node:

$$Node_{AVF}(c) = \sum_{i=1}^3 AVF_i(c) \cdot f(i) \quad (3)$$

where, $Node_{AVF}$ is the AVF for each technology node, c is the component, i is the number of injected faults (1, 2, and 3 in our case), and f is the fault rate of each class.

The calculated AVFs for every technology node are shown in Fig. 7. Each bar shows the estimated AVF, separated in 2 colors. Green corresponds to single-bit AVF while red color shows the component's AVF difference when multi-bit faults are also considered. For each technology, the single-bit only AVF is the same as the 250 nm AVF because this is the only node with only single-bit upsets. The difference of each technology bar (red color) shows the *assessment gap* that our analysis measures for each component: i.e. the *actual* AVF when all realistic multi-bit upsets are considered vs. the dry single-bit upset only AVF. This varies from 11% (DTLB) to 35% (register file) AVF difference for 22nm. As expected, the AVF of more recent nodes is significantly higher, due to the fact that the ratio of MBUs is higher and, in every single case, the AVF of MBUs was estimated higher compared to single-bit fault AVF. As the ratio of MBUs is the same for all components in a particular technology, the

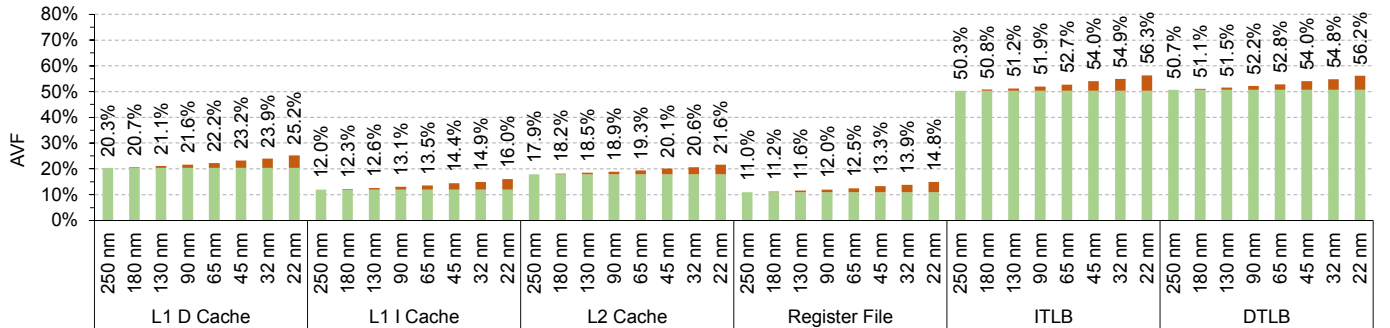


Fig. 7. Multi-bit upsets weighted AVF per component for different technology nodes. Green color corresponds to single-bit fault vulnerability, which matches the vulnerability of 250nm, while red color illustrates the vulnerability due to multi-bit upsets. For example, in the L1I cache when only single bit upsets are considered, the AVF would be only 12% (the AVF of the 250 nm) while the actual multi-bit AVF in the 22 nm node is 16%; a very large 33% difference that single-bit assessment (through injection or other methods) misses.

TABLE VII. RAW FIT FOR 250NM TO 22NM NODES.

Node	Raw FIT per bit
250 nm	47×10^{-8}
180 nm	85×10^{-8}
130 nm	106×10^{-8}
90 nm	100×10^{-8}
65 nm	85×10^{-8}
45 nm	58×10^{-8}
32 nm	38×10^{-8}
22 nm	23×10^{-8}

TABLE VIII. COMPONENT SIZES IN BITS.

Component	Size (in bits)
L1D Cache	262144
L1I Cache	262144
L2 Cache	4194304
Register File	2112
ITLB	1024
DTLB	1024

trends that existed in each component remain unchanged. The highest vulnerability is reported for the ITLB at 56.3% AVF for the 22 nm node. This means that typical reliability estimations that only consider single bit faults would miss as much as 6 percentile units (i.e., 11.9% loss) in the AVF estimation. This applies to all of the studied components, with the minimum deviation observed in L2 cache, at 3.6 percentile units (i.e., 20.2% loss). We can also observe that the AVF difference of every step is larger compared to the previous steps (e.g., the difference from 180nm to 130nm is larger than the difference from 250nm to 180nm) with an exception of the 45nm to 32nm step, in which the rates are always smaller than the 65nm to 45nm case.

VI. FAILURES IN TIME ANALYSIS PER TECHNOLOGY NODE

Failures in Time (FIT) rate of a device is the number of failures that can be expected in one billion (10^9) device-hours of operation. For each hardware structure in a microprocessor, a different FIT is computed using the formula in Eq. 4 below. As we can see, the FIT of the structure is affected by three components: the FIT_{bit} (or raw FIT) rate, which is determined by the fabrication technology (Table VII) and expresses the fault rate of a single bit, the number of bits of the structure and the AVF of the structure, which is affected by the microarchitecture and the running workload. The raw FIT rate expresses the number of soft-errors that will be introduced in the component, while the AVF is the derating factor that quantifies how many of these upsets will lead to a failure. The product equals to the FIT rate of a component. The FIT of the entire CPU is calculated by adding the individual FITs of the structures.

$$FIT_{struct} = AVF_{struct} \times rawFIT_{bit} \times \#Bits_{struct} \quad (4)$$

For each technology node, we calculate the FIT of the core by adding the corresponding FIT of all components. For the calculation of the FIT of a component for a specific technology node, we use the rawFIT rate per bit of Table VII [1]. Notice that the per-bit FIT rate increases from 250 nm to 130 nm and then it starts to decrease. Although the devices become more sensitive, the effects of the high-density overpass the decreased reliability, when projected in a bit-size relative format. The size of each of the six components is listed in Table VIII.

Fig. 8 shows the total CPU FIT rate for each technology node using our analysis findings. The red color indicates the percentage of FIT due to multi-bit faults, which starts from 0% in 250nm node and reaches a high 21% in 22nm. This is the portion that is normally ignored by single-bit fault estimations. We can also see that the FIT for each component is increasing until the

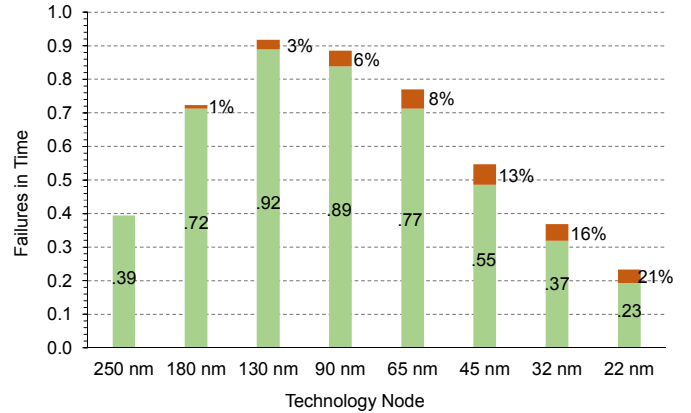


Fig. 8. FIT for the entire CPU core for different technology nodes (numbers inside the green bars). Red color areas correspond to the contribution of multi-bit upsets.

point of 130nm. After that, the FIT rate starts to decrease, reaching the lowest FIT values at 22 nm. The trend is aligned with the rawFIT rates of Table VII. These values correspond to the exact same microarchitecture with the exact same configuration. The differences observed are due to the much smaller area that the chip occupies in the higher density technologies, which results to a significantly smaller number of particles that will eventually strike the processor. On the other hand, each of these particles has a higher probability to cause multi-bit upsets and higher chances of failures.

VII. CONCLUSION

In this paper, a complete analysis of out-of-order microprocessors vulnerability to realistic spatial multi-bit upsets across different manufacturing technologies. We reported how MBUs affect the vulnerability of 6 different microarchitectural components, using an enhanced version of the GeFIN fault injector. Our analysis shows that the vulnerability is significantly higher on multi-bit faults, up to 3.2x (+220%) between single and triple-bit fault injection. By considering the ratio for MBUs in several fabrication technology nodes, we calculated the complete realistic AVFs of the hardware components considering all different types of MBUs. The results show that a difference up 35% in the vulnerability estimation was caused due to MBUs in 22nm fabrication technology. Using these results along with the soft-error rates of each technology, we calculated the overall reliability (FIT rates) of the entire CPU in 8 different technology nodes. The results show that the estimated FIT rate can be up to 21% higher when considering multi-bit faults.

The presented analysis highlights the importance of the continuously evolving problem of MBUs in microprocessors. It is generic, and as so, also applicable to other CPU models (e.g., in-order CPUs) and ISAs (e.g., x86, RISC-V) and can be performed to post 22nm technology nodes (including current FinFET nodes and forthcoming technologies) for which we expect the per-component AVF and the overall microprocessor FIT rates assessment gaps between single-bit and aggregate multi-bit faults to be even larger because of the higher rates of multi-bit faults in the CPU structures.

REFERENCES

- [1] E. Ibe, H. Taniguchi, Y. Yahagi, K. Shimbo, and T. Toba, "Impact of Scaling on Neutron-Induced Soft Error in SRAMs From a 250 nm to a 22 nm Design Rule," *IEEE Transactions on Electron Devices*, vol. 57, no. 7, pp. 1527–1538, Jul. 2010 [Online]. Available: <http://dx.doi.org/10.1109/TED.2010.2047907>
- [2] N. Binkert, B. Beckmann, G. Black, S. K. Reinhardt, A. Saidi, A. Basu, J. Hestness, D. R. Hower, T. Krishna, S. Sardashti, R. Sen, K. Sewell, M. Shoaib, N. Vaish, M. D. Hill, and D. A. Wood, "The gem5 simulator," *SIGARCH Comput. Archit. News*, vol. 39, no. 2, pp. 1–7, Aug 2011. [Online]. Available: <http://doi.acm.org/10.1145/2024716.2024718>
- [3] A. Chatzidimitriou and D. Gizopoulos, "Anatomy of microarchitecture-level reliability assessment: Throughput and accuracy," in *2016 IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS)*. IEEE, Apr 2016. [Online]. Available: <https://doi.org/10.1109/ispass.2016.7482075>
- [4] R. Leveugle, A. Calvez, P. Maistri, and P. Vanhauwaert, "Statistical fault injection: Quantified error and confidence," in *2009 Design, Automation & Test in Europe Conference & Exhibition, 2009* [Online]. Available: <http://dx.doi.org/10.1109/DATE.2009.5090716>
- [5] S. S. Mukherjee, C. Weaver, J. Emer, S. K. Reinhardt, and T. Austin, "A Systematic Methodology to Compute the Architectural Vulnerability Factors for a High-Performance Microprocessor," in *Proceedings of the 36th Annual IEEE/ACM International Symposium on Microarchitecture*. Washington, DC, USA: IEEE Computer Society, 2003.
- [6] M. Kaliorakis, S. Tselonis, A. Chatzidimitriou, N. Foutris, D. Gizopoulos, "Differential Fault Injection on Microarchitectural Simulators," *IEEE International Symposium on Workload Characterization (IISWC 2015)*, Atlanta, GA, USA, October 2015.
- [7] M. R. Guthaus, J. S. Ringenber, D. Ernst, T. M. Austin, T. Mudge, and R. B. Brown, "Mibench: A free, commercially representative embedded benchmark suite," in *Proceedings of the Fourth Annual IEEE International Workshop on Workload Characterization. WWC-4 (Cat. No.01EX538)*, Dec 2001, pp. 3–14.
- [8] N. George, C. R. Elks, B. W. Johnson, and J. Lach, "Transient fault models and AVF estimation revisited," in *2010 IEEE/IFIP International Conference on Dependable Systems & Networks (DSN)*, 2010 [Online]. Available: <http://dx.doi.org/10.1109/DSN.2010.5544276>
- [9] D. S. Khudia and S. Mahlke, "Harnessing Soft Computations for Low-Budget Fault Tolerance," in *2014 47th Annual IEEE/ACM International Symposium on Microarchitecture*, 2014 [Online]. Available: <http://dx.doi.org/10.1109/MICRO.2014.33>
- [10] A. A. Nair, L. K. John, and L. Eeckhout, "AVF Stressmark: Towards an Automated Methodology for Bounding the Worst-Case Vulnerability to Soft Errors," in *2010 43rd Annual IEEE/ACM International Symposium on Microarchitecture*, 2010 [Online]. Available: <http://dx.doi.org/10.1109/MICRO.2010.34>
- [11] Z. Zhao, D. Lee, A. Gerstlauer, L. K. John, "Host-compiled reliability modeling for fast estimation of architectural vulnerabilities", *SELSE 2015*.
- [12] J. L. Henning, "Spec cpu2006 benchmark descriptions," *SIGARCH Comput. Archit. News*, vol. 34, pp. 1–17, Sept. 2006.
- [13] R. C. Baumann, "Soft errors in advanced computer systems", *IEEE Design & Test of Comp.*, vol. 22, no. 3, pp. 258-266, May/June 2005.
- [14] Z. Chishti, A. R. Alameldeen, C. Wilkerson, W. Wu, S.-L. Lu, "Improving cache lifetime reliability at ultra-low voltages", *MICRO 2009*.
- [15] C. Constantinescu, "Trends and challenges in VLSI circuit reliability", *IEEE Micro*, vol. 23, pp. 14-19, July 2003.
- [16] S. Nassif, N. Mehta, and Y. Cao, "A resilience roadmap", *DATE 2010*.
- [17] Y. Luo, S. Govindan, B. Sharma, M. Santaniello, J. Meza, A. Kansal, J. Liu, B. Khessib, K. Vaid, and O. Mutlu, "Characterizing Application Memory Error Vulnerability to Optimize Datacenter Cost via Heterogeneous-Reliability Memory," in *2014 44th Annual IEEE/IFIP International Conference on Dependable Systems and Networks*, 2014 [Online]. Available: <http://dx.doi.org/10.1109/DSN.2014.50>
- [18] S. Pan, Y. Hu, and X. Li, "IVF: Characterizing the Vulnerability of Microprocessor Structures to Intermittent Faults," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 20, no. 5, pp. 777–790, May 2012 [Online]. Available: <http://dx.doi.org/10.1109/TVLSI.2011.2134115>
- [19] F. A. Bower, D. Hower, M. Yilmaz, D. J. Sorin, and S. Ozev, "Applying architectural vulnerability Analysis to hard faults in the microprocessor," *ACM SIGMETRICS Performance Evaluation Review*, vol. 34, no. 1, p. 375, Jun. 2006 [Online]. Available: <http://dx.doi.org/10.1145/1140103.1140327>
- [20] V. Sridharan, D. R. Kaeli, "Using hardware vulnerability factors to enhance AVF analysis", *ISCA 2010*.
- [21] V. Sridharan, D. R. Kaeli, "Eliminating microarchitectural dependency from architectural vulnerability", *IEEE International Symposium on High Performance Computer Architecture (HPCA-15)*, 2009.
- [22] N. Seifert, B. Gill, S. Jahinuzzaman, J. Basile, V. Ambrose, S. Quan, R. Allmon, and A. Bramnik, "Soft error susceptibilities of 22nm tri-gate devices," *IEEE Transactions on Nuclear Science*, pp. 2666–2673, Dec 2012.
- [23] M. Wilkening, V. Sridharan, S. Li, F. Previlon, S. Gurumurthi, and D. R. Kaeli, "Calculating Architectural Vulnerability Factors for Spatial Multi-bit Transient Faults", *IEEE/ACM International Symposium on Microarchitecture (MICRO)*, 2014.
- [24] Q. Lu, M. Farahani, J. Wei, A. Thomas, and K. Pattabiraman, "LLFI: An intermediate code-level fault injection tool for hardware faults," in *2015 IEEE International Conference on Software Quality, Reliability and Security*, 2015, pp. 11–16.
- [25] F. Ayatollahi, B. Sangchoolie, R. Johansson, and J. Karlsson, "A study of the impact of single bit-flip and double bit-flip errors on program execution," in *Proceedings of the 32nd International Conference on Computer Safety, Reliability, and Security*, ser. *SAFECOMP 2013*. Springer-Verlag New York, Inc., 2013, pp. 265–276.
- [26] F. Adamu-Fika and A. Jhumka, "An investigation of the impact of double bit-flip error variants on program execution," in *Proceedings of the 15th International Conference on Algorithms and Architectures for Parallel Processing*. Springer International Publishing, 2015, pp. 799–813.
- [27] B. Sangchoolie, K. Pattabiraman, and J. Karlsson, "One Bit is (Not) Enough: An Empirical Study of the Impact of Single and Multiple Bit-Flip Errors," in *2017 47th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, 2017.
- [28] G. Yalcin, O. S. Unsal, A. Cristal, and M. Valero, "FIMSIM: A fault injection infrastructure for microarchitectural simulators," in *2011 IEEE 29th International Conference on Computer Design (ICCD)*. IEEE, Oct 2011. [Online]. Available: <https://doi.org/10.1109/iccd.2011.6081435>
- [29] N. Foutris, D. Gizopoulos, J. Kalamatianos, and V. Sridharan, "Assessing the impact of hard faults in performance components of modern microprocessors," in *2013 IEEE 31st International Conference on Computer Design (ICCD)*. IEEE, Oct 2013. [Online]. Available: <https://doi.org/10.1109/iccd.2013.6657044>
- [30] A. Chatzidimitriou, G. Papadimitriou, D. Gizopoulos, S. Ganapathy, and J. Kalamatianos, "Analysis and characterization of ultra low power branch predictors," in *2018 IEEE International Conference on Computer Design (ICCD)*. IEEE, Oct 2018.
- [31] A. Chatzidimitriou, G. Panadimitriou, D. Gizopoulos, S. Ganapathy, and J. Kalamatianos, "Assessing the Effects of Low Voltage in Branch Prediction Units," in *2019 IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS)*, 2019 [Online]. Available: <http://dx.doi.org/10.1109/ISPASS.2019.00020>
- [32] A. Chatzidimitriou, P. Bodmann, G. Papadimitriou, D. Gizopoulos, and P. Rech, "Demystifying Soft Error Assessment Strategies on ARM CPUs: Microarchitectural Fault Injection vs. Neutron Beam Experiments", *IEEE/IFIP International Conference on Dependable Systems and Networks (DSN 2019)*, Portland, Oregon, USA, June 2019.
- [33] C.-K. Chang, S. Lym, N. Kelly, M. B. Sullivan, and M. Erez, "Hamartia: A fast and accurate error injection framework," in *2018 48th Annual IEEE/IFIP International Conference on Dependable Systems and Networks Workshops (DSN-W)*. IEEE, Jun 2018. [Online]. Available: <https://doi.org/10.1109/dsn-w.2018.00046>
- [34] R. B. Tonetto, G. L. Nazar, and A. C. S. Beck, "Precise evaluation of the fault sensitivity of OoO superscalar processors," in *2018 Design, Automation & Test in Europe Conference & Exhibition (DATE)*. IEEE, Mar 2018. [Online]. Available: <https://doi.org/10.23919/date.2018.8342082>

- [35] E. Cheng, P. Bose, S. Mitra, S. Mirkhani, L. G. Szafaryn, C.-Y. Cher, H. Cho, K. Skadron, M. R. Stan, K. Lilja, and J. A. Abraham, "Clear," in Proceedings of the 53rd Annual Design Automation Conference on - DAC '16. ACM Press, 2016. [Online]. Available: <https://doi.org/10.1145/2897937.2897996>
- [36] V. B. Kleeburger, C. Gimmler-Dumont, C. Weis, A. Herkersdorf, D. Mueller-Gritschneider, S. R. Nassif, U. Schlichtmann, and N. Wehn, "A cross-layer technology-based study of how memory errors impact system resilience," *IEEE Micro*, vol. 33, no. 4, pp. 46–55, Jul 2013. [Online]. Available: <https://doi.org/10.1109/mm.2013.67>
- [37] A. Vallero, A. Savino, G. Politano, S. D. Carlo, A. Chatzidimitriou, S. Tselonis, M. Kaliorakis, D. Gizopoulos, M. Riera, R. Canal, A. Gonzalez, M. Kooli, A. Bosio, and G. D. Natale, "Cross-layer system reliability assessment framework for hardware faults," in 2016 IEEE International Test Conference (ITC). IEEE, Nov 2016. [Online]. Available: <https://doi.org/10.1109/test.2016.7805863>
- [38] A. Vallero, A. Savino, A. Chatzidimitriou, M. Kaliorakis, M. Kooli, M. R. Villanueva, G. D. Natale, A. Bosio, R. Canal, D. Gizopoulos, and S. D. Carlo, "SyRA: Early system reliability analysis for cross-layer soft errors resilience in memory arrays of microprocessor systems," *IEEE Transactions on Computers*, pp. 1–1, 2018. [Online]. Available: <https://doi.org/10.1109/tc.2018.2887225>
- [39] N. J. George, C. R. Elks, B. W. Johnson, and J. Lach, "Bit-slice logic interleaving for spatial multi-bit soft-error tolerance," in Int'l Conference on Dependable Systems and Networks (DSN), 2010, pp. 141–150.
- [40] J. Suh, M. Annavaram, and M. Dubois, "MACAU: A Markov model for reliability evaluations of caches under single-bit and multi-bit upsets," in Int'l Symposium on High-Performance Computer Architecture (HPCA), 2012, pp. 1–12.
- [41] A. Dixit and A. Wood, "The impact of new technology on soft error rates," in 2011 International Reliability Physics Symposium, 2011 [Online]. Available: <http://dx.doi.org/10.1109/irps.2011.5784522>
- [42] G. Georgakos, P. Huber, M. Ostermayr, E. Amirante, and F. Ruckerbauer, "Investigation of Increased Multi-Bit Failure Rate Due to Neutron Induced SEU in Advanced Embedded SRAMs," in 2007 IEEE Symposium on VLSI Circuits, 2007 [Online]. Available: <http://dx.doi.org/10.1109/vlsic.2007.4342774>
- [43] M. Ebrahimi, A. Evans, M. B. Tahoori, E. Costenaro, D. Alexandrescu, V. Chandra, and R. Seyyedi, "Comprehensive Analysis of Sequential and Combinational Soft Errors in an Embedded Processor," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 34, no. 10, pp. 1586–1599, Oct. 2015 [Online]. Available: <http://dx.doi.org/10.1109/tcad.2015.2422845>
- [44] J. Maiz, S. Hareland, K. Zhang, and P. Armstrong, "Characterization of multi-bit soft error events in advanced SRAMs," in IEEE International Electron Devices Meeting 2003 [Online]. Available: <http://dx.doi.org/10.1109/iedm.2003.1269335>
- [45] M. Maniatakos, M. Michael, C. Tirumurti, and Y. Makris, "Revisiting Vulnerability Analysis in Modern Microprocessors," *IEEE Transactions on Computers*, vol. 64, no. 9, pp. 2664–2674, Sep. 2015 [Online]. Available: <http://dx.doi.org/10.1109/tc.2014.2375232>
- [46] M. Maniatakos, M. K. Michael, and Y. Makris, "Multiple-Bit Upset Protection in Microprocessor Memory Arrays Using Vulnerability-Based Parity Optimization and Interleaving," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 23, no. 11, pp. 2447–2460, Nov. 2015 [Online]. Available: <http://dx.doi.org/10.1109/tvlsi.2014.2365032>
- [47] X. Li, S. V. Adve, P. Bose, J. A. Rivers, "Scaling of Architecture Level Soft Error Rates for Superscalar Processors," Proc. 1st Workshop on the System Effects of Logic Soft Errors (SELSE), April 2005.
- [48] A. Chatzidimitriou, M. Kaliorakis, D. Gizopoulos, M. Iacaruso, M. Pipponzi, R. Mariani, S. Di Carlo, "RT Level vs. Microarchitecture-Level Reliability Assessment: Case Study on ARM(R) Cortex(R)-A9 CPU," in 2017 47th Annual IEEE/IFIP International Conference on Dependable Systems and Networks Workshops (DSN-W), 2017 [Online]. Available: <http://dx.doi.org/10.1109/DSN-W.2017.16>