# Estimating the Effect of Single-event Upsets on Microprocessors

Cristian Constantinescu*, Srini Krishnamoorthy ** and Tuyen Nguyen***
Advanced Micro Devices, Inc.
* Fort Collins, CO, USA
** Sunnyvale, CA, USA
***Austin, TX, USA
e-mail: {cristian.constantinescu; srini.krishnamoorthy; tuyen.nguyen}@amd.com

*Abstract*—**Evaluating the impact of single-event upsets (SEUs) on complex VLSI circuits in general, and microprocessors in particular, requires an interdisciplinary approach, that includes soft error modeling, accelerated measurements, derating of the raw error rates, and specialized design tools. This paper discusses modeling techniques employed to estimate the soft error rates (SER) of storage cells, provides results of accelerated measurements for three technology nodes, and presents a technique for derating the raw error rates by simulated error injection. We use the measurement results to validate and calibrate the models. Then present the tool employed for deriving the SER of the Advanced Micro Devices processor code-named "Bulldozer" and examples of estimated SER. Our approach enables the cost-effective mitigation of SEU by employing data integrity protection for the most sensitive logic.**

***Keywords: single-event upsets, soft errors, microprocessors***

## I. INTRODUCTION

Complex interconnected computing systems pervade our daily lives by performing both routine tasks and critical applications. As a consequence, the reliability of computers and networks grows increasingly important. Significant efforts have been directed towards increasing the resilience of microprocessors to single-event upsets (SEUs). SEUs are induced by energetic particles, such as neutron, protons, and heavy ions. The alpha particles emitted by packaging and interconnect materials may also lead to SEUs. Semiconductor storage cells – SRAM and sequential logic – are the most sensitive to SEUs. Energetic particles also can induce pulses in combinational logic, known as single-event transients (SETs). Errors generated by SEU and SET are referred to as soft errors and could lead to system crashes and silent data corruption (SDC) [1, 2].

The development of microprocessors resilient to soft errors is an interdisciplinary process. It requires modeling the impact of SEUs, accelerated measurements of soft error rates (SER), and cost-effective design solutions. Modeling usually is performed by circuit simulations [3, 4, 5] and technology computer-aided design (TCAD) techniques [6]. Measuring SER requires radiation sources with energetic spectrums as close as possible to the natural radiation sources. Los Alamos Neutron Science Center (LANSCE) is one of the most commonly used facilities for performing accelerated measurements of SER. LANSCE provides a neutron beam similar to the natural one, up to 600 MeV, and an acceleration factor of about $10^6$ [7].

Processor designers also must consider the fact that not all SEU-induced errors are visible to the users. For instance, some soft errors are overwritten before data is sent to memory or I/O devices, and others are masked as they propagate through the combinational logic. Several approaches, like SER derating based on error injection [8, 9] and evaluation of architectural vulnerability factors (AVF) [10, 11], were developed to derive the processor SER accurately. The complexity of today's microprocessors also requires specialized computer aided design (CAD) tools for estimating SER. Such tools bring together the results of SER modeling and measurements, the derating factors obtained by error injection and/or AVF, and the specifics of the design, like the number and type of storage cells employed by every unit/block of the microprocessor. A CAD tool developed at AMD was used to estimate SER of the AMD microprocessor code-named "Bulldozer", implemented in the 32 nm SOI process.

Mitigating the impact of SEU relies on process-level techniques, circuit solutions, and processor-level redundancy and recovery. The silicon-on-insulator (SOI) process is significantly less sensitive to SEU, compared to traditional bulk technology [12]. FinFET transistors also are expected to provide good immunity to SEU [13]. A wide variety of SEU-resilient SRAM and flip-flop (FF) designs have been proposed [14, 15, 16]. At the processor level, parity and error-detection and -correction codes (ECC) are extensively used. In addition, high-end server processors rely on checkpointing of the architectural state and instruction retry to recover from SEU-induced errors [17, 18]. Unfortunately, performance, power, silicon area, and cost penalties are associated with all these techniques. Accurate estimation of SER for every structure in a microprocessor may limit the overhead costs by concentrating protection on the most sensitive logic of the design.

This paper presents AMD's approach to estimate the impact of SEU on microprocessors. SER modeling techniques are discussed in Section II. Section III provides results of neutron accelerated measurements both for SOI and bulk technologies. Techniques for derating SER, as well as results

978-1-4799-6155-9/14/$31.00 ©2014 IEEE

of error injection experiments, are discussed in Section IV. The CAD tool used for estimating the SER of Bulldozer is presented in Section V. Section VI concludes the paper.

## II. Modeling Soft Errors

The physics governing the interaction of energetic particles with the materials used for manufacturing semiconductor devices makes modeling soft errors quite complex. There are two main approaches for estimating SER: circuit-level models and TCAD-based models. Mixed-mode solutions that combined circuit- and TCAD-based models have been implemented to improve the simulation time.

Circuit-level models primarily rely on SPICE simulations [3, 4, 5]. SPICE models incorporate information about device size, parasitics, silicon thickness, gate-oxide thickness, doping levels, and material properties. The SEU-induced charge may be modeled by either a piece-wise linear [5] or double-exponential [3] current pulse. The area under the pulse represents the charge induced by the striking particle. The current source generating the current pulse is connected to an off-device of the storage cell (e.g., the pull-down transistor of a SRAM cell). The amount of charge delivered is increased gradually until the targeted node is turned on (i.e., the cell "flips"). The minimum charge that changes the state of the cell is referred to as critical charge, $Q_{crit}$. A detailed example of SPICE SEU simulation of an SRAM cell is provided in [3]. Calculating the SER of a circuit requires the flux of the energetic particles considered (e.g., the natural neutron flux) and the cross-section of the circuit. The cross-section is a function of $Q_{crit}$, diffusion area, voltage, and process parameters. As a result, SER can be expressed as:

$$SER = N*A*C*e^{-(Qcrit/Qs)} \quad (1)$$

where N is the particle flux, A is the diffusion area, C is a correction factor, and $Q_s$ is the collection slope. Both C and $Q_s$ are empirical parameters, used to calibrate the model against measurement results [4]. In the case of complex circuits, Equation (1) can be used in conjunction with the methodology described in [19].

The accuracy of SER estimates can be improved by TCAD simulations. TCAD tools employ 3D solid models of the circuit under investigation. These models accurately account for geometry and physical dimensions of the devices and the back-end-of-line (BEOL) interconnects. Researchers at Vanderbilt University developed the Monte Carlo Radiative Energy Deposition (MRED) tool for modeling nuclear events. The tool is based on the Geant4 radiation transport library [6, 20]. The events generated by MRED were used as inputs for the Synopsys Sentaurus TCAD tool [21]. This approach was used for studying the impact of heavy ions on a 10-transistor radiation-hardened SRAM cell [20], in which a 3-D solid model of four PMOS and two pull-down NMOS transistors was created. The remaining circuitry of the SRAM cell was simulated in SPICE to shorten the run-time. SEU analysis of a much larger circuit, a 64-transistor radiation-hardened FF, is presented in [6]. In this case MRED generated events were conveyed automatically to an HSPICE simulator. The charges generated by ions crossing the solid model were converted into double-exponential current pulses, which were injected into the circuit netlist. The results of current injection were analyzed to find SEUs, and calculate SER.

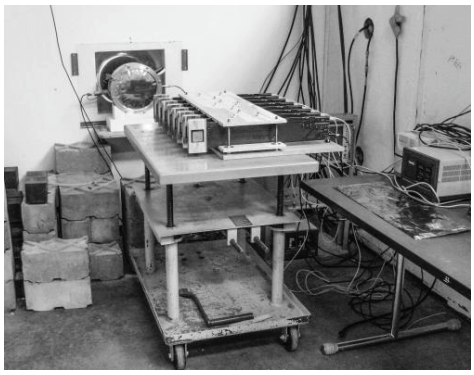


Figure 1. SER accelerated measurement set-up at LANSCE

The SER calculation engine of the CAD tool described in Section V employs Equation (1) and methodology described in [19]. Guidance on the size and shape of the injected current pulses was obtained from mixed-mode TCAD and HSPICE simulations conducted as described in [20].

## III. Accelerated Measurements

SER measurements are a vital part of any SER program. First, measured SER is used for validating and calibrating circuit-level and TCAD-based models. Second, measurements provide data that is hard to obtain from simulations, like multi-bit error patterns and rates. Multi-bit error data is necessary to select the codes used for detection and correction of soft errors and interleaving which is employed to avoid more than one error in a code word. Third, radiation-hardened designs require extensive accelerated testing. Fourth, measurements are necessary to assess technology scaling trends, the impact of voltage scaling, and the effect of process changes. For instance, neutron measurements were used to validate the SER model proposed in [4]. Measured multi-bit error rates and neutron SER sensitivity to voltage scaling were reported in [22]. Seifert et al. measured both neutron and alpha SER of tri-gate SRAM and sequential logic, and compared it to planar bulk SER. These measurements showed that 22 nm tri-gate devices had significantly lower SER: 1.5 – 4x times lower neutron SER, and more than 10x lower alpha SER, compared to planar bulk SER [13].

Due to limited space, we provide only a few examples of measured sequential logic neutron SER. Accelerated measurements were performed at LANSCE. Figure 1 shows a typical experimental set-up. A number of specially designed test boards are placed in the neutron beam. Only the test chips are exposed to the beam because the control logic is placed at the other end of the boards. The experiment is monitored from an adjacent room to ensure personnel safety.

Figures 2 and 3 provide the SER of two master-slave flip-flops (MSFF), size 2 and size 8, respectively, for three technology nodes: 45 nm SOI, 32 nm SOI, and 32 nm bulk. For the SOI manufacturing process technology scaling has a positive impact: the SER of the 32 nm FF is lower than SER of the 45 nm FF (2x to 4x at 0.8V). Although $Q_{crit}$ is lower, the

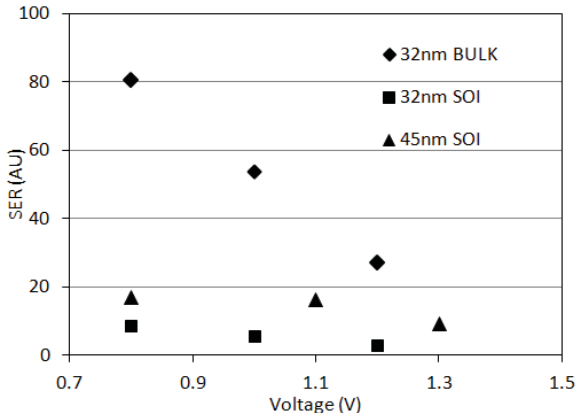



Figure 2. SER of a size 2 MSFF as function of voltage (bulk SER at 1.2V derived by linear extrapolation). AU – arbitrary units

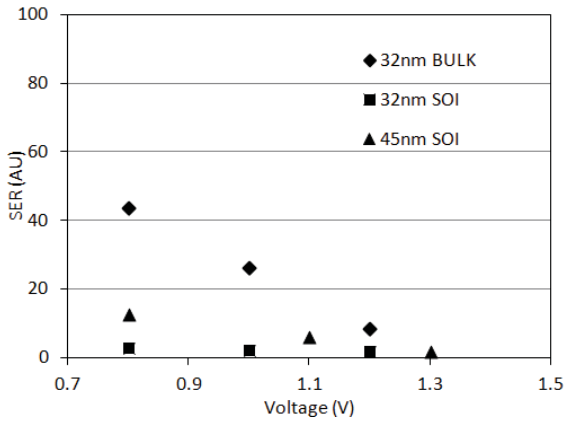



Figure 3. SER of a size 8 MSFF as function of voltage (bulk SER at 1.2V derived by linear extrapolation). AU – arbitrary units

smaller charge-collection volume of the 32 nm process has a prevailing impact.

The SER of the 32 nm bulk MSFFs is higher compared to 32 nm SOI across the voltage range for both sizes. The largest difference is at 0.8V: 10x and 8x, for sizes 2 and 8, respectively. This is explained primarily by the smaller charge-collection volume of the MSFF manufactured in SOI technology. Figures 2 and 3 also show the impact of voltage scaling. For all technology nodes lower voltage correlates to higher SER ($Q_{crit}$ is lower at lower voltages). The SER slope is very steep for both sizes 2 and 8 bulk MSFF, while SOI MSFFs show a lower SER-voltage dependency. Across the voltage range, the SER of the size 8 MSFF is significantly lower than the SER of the size 2 MSFF, primarily due to the driver strength and larger latch keepers.

The results of the preceding SER measurements were used to calibrate the $Q_{crit}$ based SER calculation engine of the CAD tool (i.e., deriving C and $Q_s$ parameters in Equation (1)).

## IV. Derating SER

As mentioned in the introduction, not all soft errors have a negative impact on the operation of the processor. In this section we discuss two different approaches for estimating the real impact of soft errors: AVF and logic derating and masking (LDM). Practically, the user-seen SER is calculated as the product of nominal SER (derived by modeling or measured) multiplied by derating (AVF or LDM).

AVF can be derived in early stages of the development process with Little's law. The average number of bits in a structure, average bandwidth of bits per cycle into that structure, and average residence time per bit in the structure are the input parameters required to estimate AVF [10, 11]. When performance simulators are available, the number of architecturally correct execution (ACE) bits crossing a structure can be counted. In the example provided in [11], the performance model approach gives AVF values in the 14 - 47% range for SPEC CPU 2000 benchmarks exercising an Intel Itanium 2-like instruction queue. The main advantages of the AVF approach are early availability of performance simulators and a short execution time. However, performance models do not include small storage structures, detailed sequential logic in the pipeline, and gate-accurate combinational logic.

When the RTL model of the processor is available, simulated error injection can be performed to estimate the impact of soft errors. Error injection provides more accurate derating estimates, because the workloads are executed on an accurate gate-level RTL model. However, RTL simulations require significant computing resources. In the rest of this section, we briefly present the methodology used for deriving SER derating of the AMD Bulldozer processor. Details on the processor architecture are given in [23]. Sensitivity to SEU, exhibited by different units and blocks, is described by LDM factors. LDM nomenclature is used to emphasize that this approach accounts for both the architectural derating and masking of the propagating errors by combinational logic.

Custom C++ software was developed for performing simulated error injection in the Verilog HDL model environment. The first step of the experiment consists of choosing the processor unit and block into which to inject errors. Then a set of workloads is selected for exercising that particular unit/block. Error-free runs are performed for each workload to determine simulation duration and status.

The second step is to generate a signal list. This list contains all storage cells that are part of the targeted unit/block (SRAM and FF lists can be generated separately). An error injection experiment consists of randomly selecting a signal from the previously generated list, reading the content of the storage cell at a random time instance, inverting the data, and writing it back (read-modify-write). In this way, a SEU is modeled accurately. The execution of the workload continues until an error is detected or until the predefined maximum duration of the simulation elapses. The LDM factor is calculated as the ratio of detected errors over the total number of injected errors.

Error injection was performed in the sequential logic of the Bulldozer microprocessor units and blocks (about half a million MSFF and latches are used in this processor).

TABLE I. EXAMPLES OF LDM FACTORS

| Unit | Block name LDM (%) | | |
|---|---|---|---|
| ID | ICTAS | ICCTL | ICTOP |
| | 10.6 | 4.5 | 0.6 |
| EXSC | SCMAP | SCAGE | SCPIC |
| | 13.0 | 12.2 | 9.4 |
| FP | FPRET | FPSCE | FPRFH |
| | 16.7 | 4.2 | 1.4 |

Benchmark traces and synthetic workloads were executed. Examples of LDM factors for several Bulldozer blocks are provided in Table I. Within the instruction fetch and decode (ID) unit, the ICTAS block has the highest LDM factor. This block receives the instruction physical address and finds whether the instruction resides in the instruction cache or the prefetch buffer. ICCTL shows an average LDM value. This is a control block processing fetch requests. It includes a prefetch buffer and a prediction queue. ICTOP primarily consists of test logic and is used very little in the normal operation of the processor. As a result, the LDM is quite low. SCMAP is the most sensitive block in the execution and scheduling (EXSC) unit (integer units are part of EXSC too). SCMAP primarily performs register renaming. A very similar LDM is shown by the SCAGE block, which attempts to select the older operations for execution. The SCPIC block, which includes the scheduler, has a slightly lower LDM factor. The most sensitive block in the floating pint (FP) unit is FPRET, the retire block, which includes the retire queue. The unit scheduler, FPSCE, has an average sensitivity. The four banks of register files within the FPRFH show a low LDM.

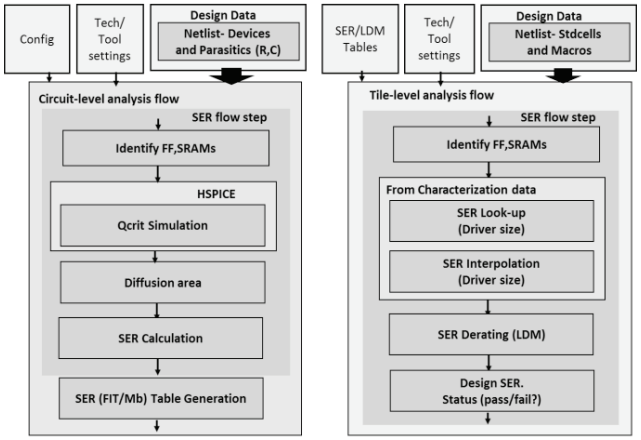
Results of error injection are valuable for cost-effective implementation of the error-detection and -correction mechanisms used by the processor, as well as for insertion of error-tolerant MSFF. For instance, most registers in the ICTAS, SCMAP, and FPRET blocks are protected by parity, while no protection is necessary in the ICTOP block. Additional details on the error injection methodology, and extensive LDM analysis for the Bulldozer processor are reported in [9].

## V. CAD TOOL FOR SER ESTIMATION

This section describes the CAD flow for SER estimation. The general approach is to pre-characterize the storage elements and look up the SER data when evaluating designs at higher levels of abstraction. Tables containing the LDM factors and other context-specific information also are used to compute the design-level SER.

### A. *Overview of the SER Estimation Flow*

At the physical-implementation level, the functional units of the processor are divided into tiles that can be either abutted or interconnected using grout logic. A tile-level design is built using the standard synthesis place-and-route (SAPR) flow and is composed of standard cells and custom macros. The SER of the tile-level design is computed as the sum of the SER of the constituent standard cells and custom macros after derating for logic-masking effects.




4(a). Circuit-level flow 4(b). Tile-level flow

Figure 4. CAD SER estimation flow

Figure 4 presents an overview of the SER estimation flow. In the characterization stage (Figure 4(a)), the extracted netlist, configuration files, and technology and tool settings data are passed as input to the SER flow. The FFs, SRAMs and other storage nodes are identified and HSPICE simulations are performed to derive $Q_{crit}$ of the nodes. The diffusion area and $Q_{crit}$ are used to compute the SER of the nodes as described in Section II. A table containing the SER of the cells and macros then is created.

Figure 4(b) shows the tile-level SER estimation flow. This flow takes as inputs the design data composed of standard cells and custom macros (output of the RTL synthesis), SER tables generated during the characterization stage, the LDM derating table based on error injection analysis, and technology and tool settings data. The sequential cells and SRAMs then are identified and their SER is looked up in the SER table. The SER of the cells that are missing in the table is derived by size-based interpolation of the available SER. Derating from the LDM table then is applied. The SER of the design is computed as the sum of derated SER of individual cells.

### B. *Circuit-level SER Computation*

For SER computation, we employ a simulation flow similar to the one used for circuit-stability analysis. In circuit-stability analysis, the goal is to assess whether a memory element can read, write, and hold the charge in its storage node(s) within a specified period [24]. In the case of SER analysis, the goal is to assess whether the state of a memory element can be disturbed by a certain level of injected charge. Consequently, in practice, these two types of analyses can share the same flow infrastructure for setting up the underlying circuit simulation.

At the cell level, the SER flow utilizes the same configuration file for each standard cell to identify the storage nodes and to set up the logic activation for state holding at each storage node. The configuration file provides the input assignments (i.e., logic activation to write into or to read from the cell and to hold the state, used for various stability simulations of the cell). For the SER simulations, we are interested in the logic activation for the hold state. Then the SEU impact is modeled as a current source, allowing us to

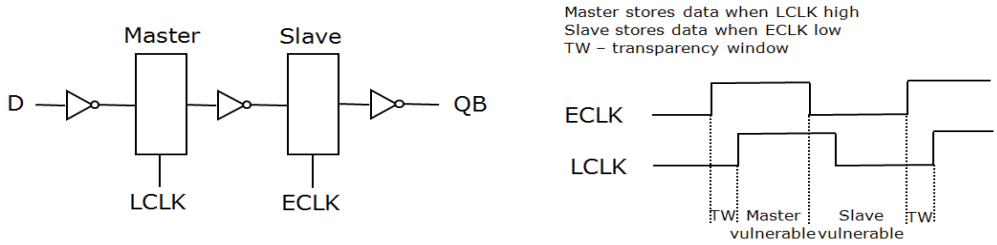



Figure 5. MSFF along with the timing diagram

determine the charge required to change the state of the cell.

For custom cell design, the transistor-level netlist typically is partitioned into channel-connected regions [25]. Then various logic structures are derived by a circuit topology recognition engine using a library of templates. The topology recognition engine also identifies the storage nodes as well as the inputs and outputs of the memory structures. The SER flow uses this information to set up the circuit simulations for deriving the $Q_{crit}$ of the custom storage nodes.

For calculating $Q_{crit}$, the injected charge is modeled as a current source in HSPICE using a double-exponential pulse [26]:

$$I(t) = I_A (e^{-t/\tau f} - e^{-t/\tau r}) \quad (2)$$

where $I_A = Q/(\tau_f - \tau_r)$ is the amplitude of the current pulse delivering the injected charge Q, and $\tau_r$ and $\tau_f$ are the rise and fall time parameters, respectively. For a given technology, appropriate ranges are determined for parameters of Equation (2) with the aid of TCAD simulations. Then a search procedure in a two-dimensional space is performed. The dimensions of this space are the amplitude and fall time of the current pulse (for the sake of simplicity, the rise time is held constant). At each point in this space, an HSPICE simulation is performed to determine whether the state of the injected node has changed. The extracted netlist used for running the HSPICE simulations also provides the diffusion area of each node. Then the SER of the node is derived using the model given by Equation (1). This SER flow is used to compute the SER values of the storage nodes, such as the master and slave nodes of a FF or the nodes of latches. The procedure to derive the overall SER for FF and latches is described in the next section.

### C. Tile-level SER Computation

The design data at this stage is generated by the RTL synthesis step and is composed of standard cells and custom macros. The SER of the cells and macros is provided by the characterization step described in Section V.B. The storage elements in the design are identified and their SER values are looked up from the characterization table. Interpolation may be used for deriving SER of the cells missing characterization data.

Because the LDM factors in the derating table were computed using error injection in the RTL representation of the design, the names of the RTL blocks in the derating table need to be mapped to post-synthesis instances. Mapping generated by logic equivalency checks (LEC) is used to identify the correct derating factors to be applied to the SER of each instance in the design.

The SER of a MSFF depends on the SEU vulnerability of the master and slave and the transparency window (TW). Figure 5 shows a MSFF along with the timing diagram of the early clock (ECLK) and late clock (LCLK). The master and slave are vulnerable to SEU as indicated in the timing diagram. Neither the master nor the slave is vulnerable while it is transparent (not storing data).

Because the SER of the master and slave storage nodes is available from the SER characterization table, the SER of the MSFF is computed as follows:

$$SER(MSFF) = SER(M) * V_m + SER(S) * V_s \quad (3)$$

where SER(M) and SER(S) are SER of the master and slave, respectively, and $V_m$ and $V_s$ are the percentages of time the master and slave are vulnerable .

The vast majority of the design's MSFF work in power-saving mode, employing clock gating. As a result, most of the time the data is held by the slave (i.e., the slave is vulnerable to SEU) and the SER of the MSFF provided by Equation (3) is adjusted:

$$SER(MSFF)_{PS} = SER(MSFF) * (100 – PS) + SER(S) * PS \quad (4)$$

where PS is the percentage of time spent in power-saving mode. For latches, the SER is computed using the SER model parameters of a typical master node. Because the latch is considered 50% transparent, the latch SER is half of the nominal SER.

The user-visible SER of each instance is calculated as:

$$SER(i) = SER_{Cell(i)} * LDM(i) \quad (5)$$

where SER(i) is the SER of instance i, Cell(i) the stdcell/macro representing the instance, and LDM(i) the derating factor of instance i. For instance, in the case of a MSFF employing power savings, $SER_{C(i)}$ is provided by Equation (4).

The tile-level SER is computed as the sum of the derated SER of all storage nodes in the tile (provided by Equation (5)). Each tile is assigned a SER budget based on an architecture-level evaluation. If the total SER of the tile is within the given budget, then the SER step is flagged as pass. If not, it is flagged as fail, indicating that either error-tolerant flops have to be swapped in, into non-critical timing paths, or that architectural changes are required to improve the SEU resilience.

Table II provides several examples of normalized tile-level

SER for the Bulldozer processor, derived by using the described CAD tool. In the ID unit, BP0 contains branch-prediction logic, while DE0 and DE1 are the decode units. In the EXSC unit, SC0 contains rename and retire-queue logic, SC1 includes the scheduler, and the execution logic is part of EX0. In the FP unit, FPFEL0 and FPFDH0 contain high-precision multiply and accumulate logic. Register files are part of FPRFI0. The unit-level SER is derived by summing the SER of all tiles in the unit. Finally, the estimated microprocessor SER is obtained by adding SER of all constituent units.

TABLE II. EXAMPLES OF NORMALIZED TILE-LEVEL SER

| Unit | Tile name<br>Normalized SER | | |
|---|---|---|---|
| ID | BP0 | DE0 | DE1 |
| | 2.46E-01 | 1.90E-01 | 1.33E-01 |
| EXSC | SC0 | SC1 | EX0 |
| | 4.25E-01 | 7.57E-01 | 3.34E-01 |
| FP | FPFEL0 | FPFDH0 | FPRFI0 |
| | 3.04E-01 | 1.76E-01 | 4.01E-01 |

## VI. CONCLUSIONS

Traditionally, the impact of SEU on complex VLSI circuits was mitigated by process techniques, error-resilient circuit designs, error detection and correction codes, and block- unit-level redundancy. However, these techniques add significant performance, power, area, and cost overhead. Accurate SER estimations allow designers to protect the most sensitive logic against SEU, minimizing the overhead. The complexity of modern microprocessors requires an interdisciplinary approach to estimate SER. Circuit-level and TCAD-based models of SRAM and sequential logic are calibrated with the aid of accelerated measurements. The same measurements provide information on multi-bit error patterns and rates of occurrence, impact of process and voltage scaling, and effectiveness of error-tolerant designs.

The sensitivity of different logic blocks to SEU is derived by RTL error injection, and represented by SER derating factors. The SER estimation of a new design is performed by a specialized CAD tool that accounts for each storage cell and merges the results of modeling and error injection. Examples of derating factors and SER estimates for AMD Bulldozer processor were provided. The approach described in this work enables the development of a SEU resilient processor without significantly increasing development or manufacturing costs.

## REFERENCES


[1] H. Ando et al., "Validation of Hardware Error Recovery Mechanisms for the SPARC64 V Microprocessor", in Proc. IEEE International Conference on Dependable Systems and Networks, 2008, pp. 62-69.

[2] C. Constantinescu, "Neutron SER characterization of microprocessors", in Proc. IEEE International Conference on Dependable Systems and Networks, 2005, pp. 754-759.

[3] M. Alles, "SPICE Analysis of the SEU Sensitivity of a Fully Depleted SO1 CMOS SRAM Cell", IEEE Trans. Nuclear Science, vol. 41, no. 6, Dec. 1994, pp. 2093-2097.

[4] P. Hazucha and C. Svensson, "Impact of CMOS technology scaling on the atmospheric neutron soft error rate", IEEE Trans. Nuclear Science, vol.47, no. 6, Dec. 2000, pp. 2586-2594.

[5] A. Maheshwari, I. Koren, and W. Burleson, "Accurate estimation of soft error rate (SER) in VLSI circuits, in IEEE International Symposium on Defect and Fault Tolerance in VLSI Systems, 2004, pp. 377-385.

[6] K. M. Warren et al., "Integrating Circuit Level Simulation and Monte-Carlo Radiation Transport Code for Single Event Upset Analysis in SEU Hardened Circuitry", IEEE Trans. Nuclear Science, vol.55, no. 6, Dec. 2008, pp. 2886-2894.

[7] S. Wender, "Accelerated neutron testing of semiconductor devices", LANSCE Activity Report, 1995-1998, pp. 52-53.

[8] H. Nguyen and Y. Yagil, "A systematic approach to SER estimation and solutions", in Proc. IEEE International Reliability Physics Symposium, 2003, pp. 60-70.

[9] C. Constantinescu, M. Butler, and C. Weller, "Error Injection-based Study of Soft Error Propagation in AMD Bulldozer Microprocessor Module", in Proc. IEEE International Conference on Dependable Systems and Networks, 2012, pp. 1-6.

[10] S. Mukherjee, C. Weaver, J. Emer, S. Reinhardt, and T. Austin, "A systematic methodology to compute the architectural vulnerability factors for a high-performance microprocessor," in Proc. International Symposium on Microarchitecture, 2003, pp. 29-40.

[11] A. Biswas, R. Cheveresan, J. Emer, S. Mukherjee, and R. Rangan, "Computing architectural vulnerability factors for address-based structures," in Proc. International Symposium on Computer Architecture, 2005, pp. 532-543.

[12] E. H. Cannon et al., "SRAM SER in 90,130 and 180 nm bulk and SOI technologies", in Proc. IEEE International Symposium on Reliability Physics, 2004, pp. 300-304.

[13] N. Seifert et al., "Soft error susceptibilities of 22 nm tri-gate devices", IEEE Trans. Nuclear Science, vol.59, no. 6, Dec. 2012, pp. 2666-2673.

[14] M. Zhang et al., "Sequential element design with built-in soft error resilience", IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 14, no. 12, 2006, pp. 1368–1378.

[15] S. Yoshimoto et al., "Multiple-bit-upset and single-bit-upset resilient 8T SRAM bitcell layout with divided wordline structure", in Proc. IEEE International On-Line Testing Symposium (IOLTS), 2011, pp. 151-156.

[16] S. J. Dillen et al., "Design and implementation of soft-edge flip-flops for x86-64 AMD microprocessor modules" in Proc. IEEE Custom Integrated Circuits Conference (CICC), 2012, pp. 1-4.

[17] P. Sanda et al., "IBM POWER6 Processor Soft Error Resilience", IBM Journal of Research and Development, vol. 52, no. 3, 2008, pp. 275-284.

[18] R. Kalla, B. Sinharoy, W. J. Starke, and M. Floyd, "Power 7: IBM's next-generation server processor", IEEE Micro, vol. 30, no. 2, 2010, pp. 7–15.

[19] M. Zhang et al., "A Soft Error Rate Analysis Methodology", IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 25, no 10, 2006, pp. 2140-2155.

[20] D. R. Ball et al., "Simulating Nuclear Events in a TCAD Model of a High-Density SEU Hardened SRAM Technology", IEEE Trans. Nuclear Science, vol.53, no. 4, Aug. 2006, pp. 1794-1798.

[21] Synopsys Sentaurus TCAD, http://www.synopsys.com/Tools/TCAD/DeviceSimulation/Pages/SentaurusDevice.aspx

[22] P. Hazucha and C. Svensson, "Cosmic Ray Neutron Multiple-Upset Measurements in a 0.6-um CMOS Process", IEEE Trans. Nuclear Science, Vol.47, No. 6, Dec. 2000, pp. 2595-2602.

[23] M. Butler, L. Barnes, D. D. Sarma, and B. Gelinas, "Bulldozer: an approach to multithreaded compute performance", IEEE Micro, vol. 31, no. 2, 2011, pp. 6-15.

[24] E. Grossar, M. Stucchi, K. Maex, and W. Dehaene, "Read stability and write-ability analysis of SRAM cells for nanometer technologies", IEEE Journal of Solid-State Circuits, vol 41, no. 11, 2006, pp. 2577–2588.

[25] R. E.Bryant, "Boolean analysis of MOS circuits", IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems, vol. 6, no. 4, July 1987, pp. 634-649.

[26] Q. Zhou and K. Mohanram, "Gate sizing to radiation hardened combinational logic", IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems, vol 25, no. 1, Jan. 2006, pp. 155 – 166.