Fault Injection in Mixed-Signal Environment Using Behavioral Fault Modeling in Verilog-A

[Submission]

ABSTRACT

Fault injection methods have been used for analysing dependability characteristics of systems for years. In this paper we propose a practical mixed-signal fault injection flow that is fast as well as accurate. Fault models are implemented directly into devices and other electrical elements on the circuit using behavioural fault description in Verilog-A language. So far, we described three classes of most common faults: i) Single event transients, ii) Electro-Magnetic interference and iii) Power disturbance faults. As an example for dependability evaluation, few test circuits have been prepared and the results of fault injection on those designs have been reported.

Keywords

Fault Injection, Fault Modeling

1. INTRODUCTION

Over the last several years, as chips feature sizes decrease to *Ultra Deep Sub Micron (UDSM)* technology, their susceptibility to outside faults increases, therefore in designing any system, reliability evaluation is an important step to ensure system's dependability in the future. Today, method of choice to analyse or validate this is fault injection.

Fault injection techniques can be classified into two categories: i) software-based fault injection, whereas the fault is injected in the software simulation model of the system and can be injected at circuit, logic or system level abstraction and ii) hardware-based fault injection where a physical system is used for experiment. In traditional simulation based fault injection, fault are usually injected at the same-level of abstraction as the design (mostly logic level or higher). Although the lower level fault injector (at circuit level) exists, they're not very common due to the large performance overhead of lower-level SPICE simulation.

The main advantage of simulation-based fault injection is the high controllability and observability it provides over the injection process This allows users to choose where and when the injection occurs and observe the fault behaviour and propagation through-out the system. Users can extract reliability parameters such as fault latency, derating rate, propagation and coverage for each module or for the entire system. Another advantage of simulation-based fault injection is that they can be used early in the design process for early discovery of design deficiencies.

As mentioned before, circuit level fault injection techniques sufferes from performance issues. Furthurmore, CAD tools are not well developed to perform verification tasks at this level. Standard circuit level simulators does not support extensive testbenches or verification schemes. On the other hand, RTL and gate level simulation are fast but they do not include enough details for accurate modelling of many fault models, specially for deep sub micron processes and beyond. Besides that, nowaday, each System-on-Chip may contain many analog and digital cores and traditional logic level fault injection techniques do not support analog or mixed-signal designs.

To address these issues, we developed a fault injection flow for mixed-signal environment and described three class of transient fault models. We address the performance issue of circuit-level fault injection without losing accuracy. Our proposed injection flow is as accurate as circuit-level fault injection and yet, it is as fast as logic/RTL level fault injection.

Our flow consists of logic-level and circuit-level mixed-signal cosimulation. We inject fault at circuit level. Near the fault site, where it matters and we require an accurate simulation, we incorporate SPICE simulation. We observed that most of the fault models will be manifested as an error outside of the target fault module. For simplicity and efficiency reason, we chose a design's hierarchy module as our fault grauality. When fault is manifested as an error, it is visible on the information state, so there is no need for SPICE level simulation anymore. At this state, logic level simulation have enough accuracy to carry on the fault simulation. Gence, the rest of the design is modeled and simulated in RTL level using logic simulation. With logic simulation we can compensate for the performance penalty of circuit-level SPICE simulation. More over, by using the previous interface of the design (HDL testbench or verification scripts) we ensure that our verification codes are still intact and valid during the fault injection session. Since the main fault injection process happens at circuit-level we can inject faults on Analog modules or modules that dont have the neccessary HDL description (for example SRAM, DRAMs, delayed Latches, PLLs, etc).

Main contribution of our work lies in our fault modeling approach. Nature of our mixed-signal flow allows us to incorporate analog and mixed-signal fault models which increases the accuracy of fault models. We propose the using of be-

havioral language (like Verilog-A) to model variety classes of fault models. Simpler fault models can be approximated at circuit level using available spice primitives (*macaromodeling*). For example, traditional modeling of Single Event Transient (SET) at circuit level usually consists of a current source at transistor source or drain. Although this approximation can be fairly accurate for submicron processes, however as aggressive device scaling into deep submicron level, these models are not accurate anymore. As IC device sizes continue to shrink the fault models are becoming more and more complicated and they can no longer be implemented solely using primitives. Also many fault models to be developed accurately require access to internal nodes, parameters or structure of the transistor or device model which is not available at circuit level. This motivates us to move to lower level of abstraction and directly implement our model inside the transistor or devices using behavioral description language such as Verilog-A. Verilog-A is the industry standard modeling language for analog circuits. It allows rapid development of new fault models and it is supported by many CAD vendors.

The rest of this paper is organized as follows: The next section is a fast review of selected litratures and methodologies on fault injection. Later in section 3 we describe fault models and their implementation details. In Sec. 4 we present our fault injection process in detail. Sec. 5 demonstrate experimental setup and the results of our approach. Finally, we conclude in Sec. 6.

2. PREVIOUS WORKS

Many researchers have investigated fault effects in circuits during the last decades. As the result, variety of fault injection methods have been proposed and used over the years. Readers can refer to [1, 2, 3] for a survey on fault injection methods and techniques.

As mentioned before, fault injection process can be software-based or hardware-based. The software based fault injection consists of two categories: i) software-implemented fault injection and ii) simulation-based fault injection. software impelemented fault injector includes [4, 5, 6, 7]. In simulation-based fault injection, faults are injected in a simulation model of the circuits. Simulation based fault injecttion can be made either at the electrical (SPICE) or logic (HDL) level or higher depending on the design's abstraction level. Fault's can be injected with means of additional elements (*saboteurs*), via alteration of simulation model (*mutants*) or through built-in simulator commands. Some of the more popular simulation-based fault injector tools are [8, 9, 10, 11, 12]. On the other hand, hardware-based injector like [13, 14, 15] inject fault in pin-level or [16, 17] stress the hardware by inducing soft errors with heavy-ion radiation.

Simulation-based fault injection tool address different abstraction level with high observability and controllability, therefore they're becoming more and more popular. These methods are usually developed for digital circuits and work at logic level of abstraction or higher, as the result, they are not much suitable for analog or mixed-signal environment. One of the early work on mixed-signal fault injection is [18]. Other example of circuit-level fault injection tools are [19, 20, 21] and [22].

3. BEHAVIORAL FAULT MODELS

Fault models are anomalies in the system and hence, they inherently cannot be accurately described with the means of standard electrical primitives. As CMOS technology continues to shrink, fault models are becoming more and more complicated and fault modeling in circuit or netlist level does not provide the required fidelity. As the result, fault injection is becoming harder, less accurate and in some cases impossible to perform. Designers are usually forced to approximate fault behaviors to be able to add their effect to the circuit and in doing so, they would reduce precision. For a reliability researcher or engineer, it is very important to have a precise description of faults and be able to rapidly model and implement new fault models as the design or technology evolve.

To develop an accurate fault model, we describe the behaviors of fault and integrate them within the device model or as a standalone element. Verilog-A language provides an accurate, yet reasonably fast way to design a fault model and inject it directly into electrical elements. Fault can be embedded inside Verilog-A module and the resulted faulty element can be attached to circuit netlist as a mutant using an external circuitry (X). With this approach, We can explore variety of fault models. Currently we've developed the fault classes:

- Single Event Transients/Upset (SET/SEU)
- Electro-Magnetic/Radio-Frequency Interference (EMI)
- Power Supply/Power Line Disturbance (PLD)

3.1 Single Event Transients/Upset (SET/SEU)

When a radioactive particle such as neutron or alpha particle strikes a sensitive region in a semiconductor device like transistors, the resulting electron-hole pair generation can cause a transient current pulse that may alter the logical state of that circuit node [23]. This transient current may consequently propagate to memory elements or primary outputs and eventually, lead to erroneous circuit behavior and cause system failure.

Over the years, many approaches have been proposed to model charge deposition current. Two of the more popular models are single and double exponential function as shown in equation 1 [24, 25]. Here τ_α is the charge collection constant of pn-junction, τ_β is the time constant for establishing the electron-hole track. An alternative model is single exponential model [24] shown in equation 2. Q denotes to the collected charge and τ is the pulse-shaping collection time parameter. Here K is a constant and equals to $\frac{2}{\sqrt{\pi}}$. Behavioral description of this model is given in Code 1.

$$I(t) = \frac{Q}{\tau_\alpha - \tau_\beta} \times (e^{-\frac{t}{\tau_\alpha}} - e^{-\frac{t}{\tau_\beta}}) \quad (1)$$

$$I(t) = \frac{K \times Q}{\tau} \times \sqrt{\frac{t}{\tau}} \times e^{-\frac{t}{\tau}} \quad (2)$$

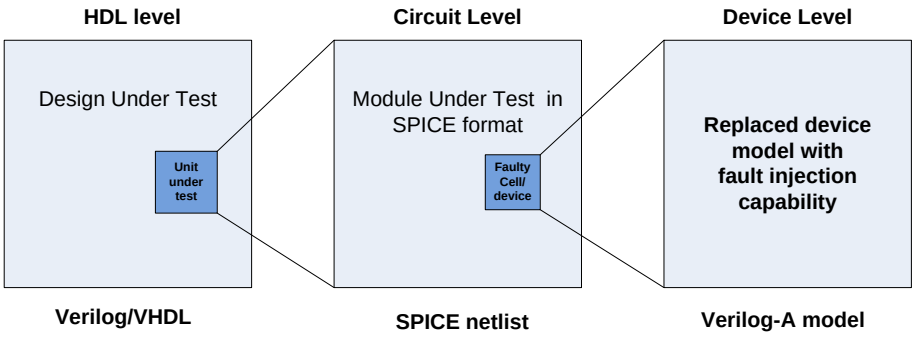

Figure 1: generic fault injection flow

```
module mosfet(drain, gate, source, bulk);
   inout drain, gate, source, bulk;
   electrical drain, gate, source, bulk;
   ...

   parameter real TYPE     = 1.0 ;
   parameter real TINJECT  = ***e-9 ;
   parameter real Q        = ***e-12 ;
   parameter real TO       = ***e-9 ;
   parameter real TB       = ***e-9 ;

  analog
   begin
   @(initial_step) ...

    ... // BSIM Transistor Equations

    `ifdef SET_MODEL_I
     if( $abstime > TINJECT )
      begin
        $strobe(SET INJECTED, @ %e
               TINJECT=%e¨ , $abstime, TINJECT);

         //double exponential SET model
        I(drain, bulk ) <+ TYPE * Q/(TO-TB)
            *(exp(-1*(($abstime-TINJECT)/TO))
            -exp(-1*(($abstime-TINJECT)/TB)));
          end
    `endif

    `ifdef SET_MODEL_II
      if( $abstime > TINJECT )
      begin
        $strobe(SET INJECTED, report @ %e
               TINJECT=%e¨ , $abstime, TINJECT);

        //single exponential SET model
        I(drain, bulk ) <+ TYPE *  1.12838*(Q/TO)
            *sqrt( ($abstime-TINJECT)/TO)
            *exp(-1*(($abstime-TINJECT)/TO));
      end
   `endif
  end
endmodule
```

Code 1: Behaviroal model of SET in Verilog-A

3.2 Electro-Magnetic/Radio-Frequency Interference (EMI)

Electromagnetic interference (*EMI*) refers to any type of interference that can potentially disrupt or degrade functioning of electronic systems. Modern integrated circuits face potential failures from electromagnetic disturbances. It is important to study and model the behavior of ICs in hostile EMI environment. Lightings, voltage lines, radar or radio transmission, wireless network and GSM bursts are usual source of EMI. In System-On-Chip components which includes RF IP cores similar problems exists as well, because of the RFI generated form the operation of RF circuitry can reach baseband circuit and induce failure. Electronic interconnects, cables or PCB traces can couple EMI in the system. Also, inside the chip metal interconnect like power, ground or clock network works as receiving antennas [26] and can induce current (mA) into the system.

Privously, [27] studied the behaviour of transistors and other electrical elements and their I-V characteristic changes under EMI. [28] studies EMI effects on CMOS devices, specifically, CMOS op-amp. They assume that EMI-induced distortion phenomena are limited to the input differential pair and model it as a continuous-wave (CW) RFI superimposed on the nominal input signals (which forces the transistors to periodically switched off by large RF disturbances). However, EMI may be injected from several nodes. For example, through the power supply lines, through the substrate node, through the clock network or the input terminals.

We simulate EMI by injection of a pulse modulated high-frequency RF signal on selected pins (like clock or data pins) or nodes inside the circuit (substrace, nodes with antenna behavior, power network, clock network, etc.).

3.3 Power Supply/Power Line Disturbance (PLD)

As we scale down the supply voltage of chips, the noise margin will decrease, hence circuit become more susceptible to power faults. Power supply disturbance are either caused by environmental disturbance (e.g. EMI/EMC) or by circuits internal switching activity. These disturbances can affect both logic and the supporting infrastructure like power grid and clock tree.

In deep-submicron processes, common power supply related

transient fault models includes power-supply noise [29, 30], power supply voltage overshoot undershoot and ground bouncing [31]. All of these faults require SPICE simulation for modeling therefore analysis of such fault models are very time consuming.

We can analyze the influence of power supply voltage disturbances and evaluate the fault tolerance of circuits with means of fault injection. We inject fault at circuit power grid. Instead of modeling a distributed power grid with RC network we use a unified Verilog-A model for the entire power network and describe the behavior of power supply disturbances into this model.

4. FAULT INJECTION FLOW

This section describes the proposed fault injection flow. Our fault injection flow is an integrated framework that provides facilities to conduct fault injection studies. In desinging this flow, accuracy, practicality and usability were our main concerns.

Transient faults can be injected into any of the locations inside the target module. The timing of the injection is configured as a parameter in fault model. The selection of when and where of the fault injection are either chosen by the user or based on randomized injection. Variety of fault models have been developed in Verilog-A and are inserted to Electrical elements. These faults are inserted to the netlist. Circuit behavior is observed through simulation trace. For each fault injection, the reliability parameters such as fault latency, derating rate, propagation and coverage are extracted. After that, a report containing statistics of experiment is generated.

This flow consists of three independent phases: I) fault injection phase, II) fault simulation phase and III) evaluation phase. Figure 2 shows an overview of our fault injection flow.

- **fault injection:** Fault injection is the foundametal step of the flow. First, we select a fault site within the design, for example, inside a processor design, a fault site can be its alu, register file SRAM, pll, or any other module. Fault site will be simulated in SPICE. This module is then converted to its equivalent circuit level netlist (through synthesizing to gate-level and netlist substition for digital module or directly replacing the analog parts for circut netlist). The resulted netlist is analyzed for extracting potential fault locations and times for injection experiments. User should provide fault model, total number of fault injection experiment to perform and define the criteria to when finishing the experiments.

  Later we inject a faulty transistor, element or other faulty components which are implemented using verilog-A directly into this circuit-level netlist as an external subcircuit or the X element. User will select the fault model from provided fault library. To save performance, we try to keep the total number of inserted Verilog-A components as minimum as possible. To do so each netlist should contain a minimal number of fault injector (perferably one). For this reason, to

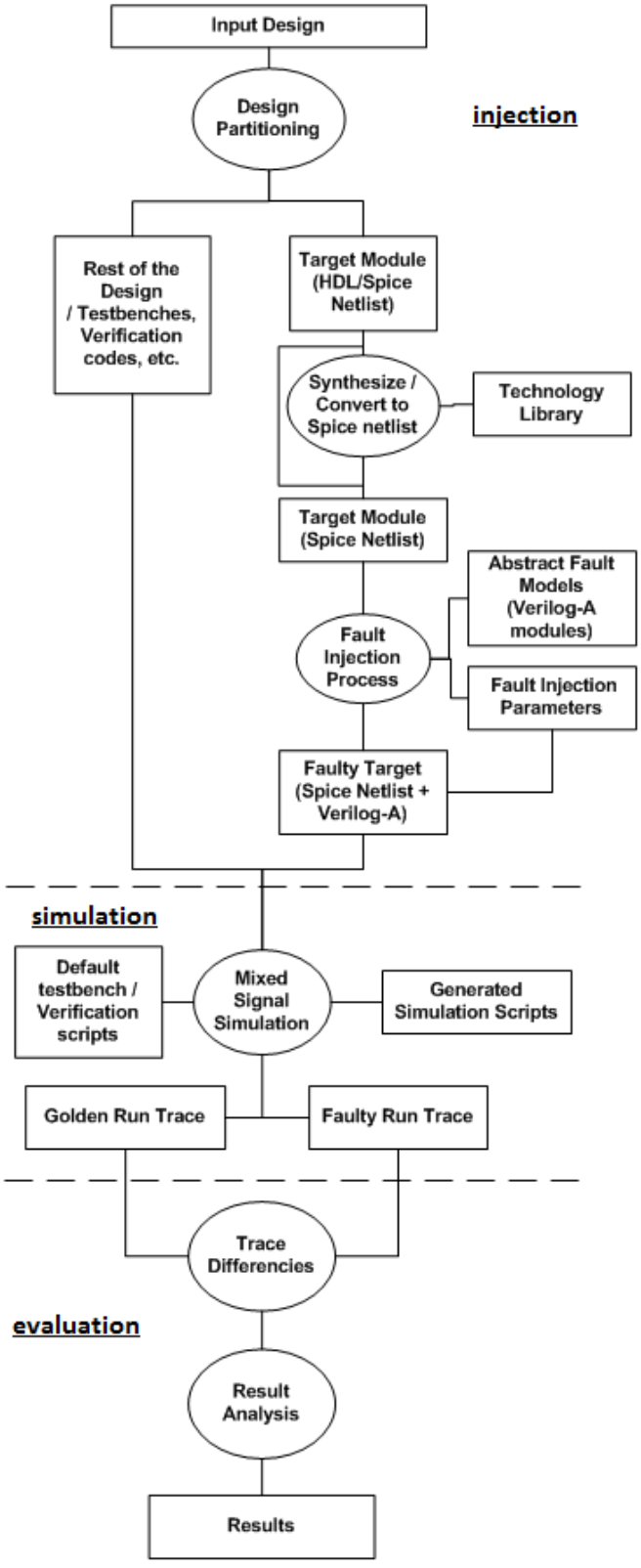

Figure 2: generic fault injection flow

perform fault injection at two different location in the netlist, we regenerate the netlist each time with specified Verilog-A injector. Flow will automatically handle this. After the target netlist is created the tool generate necessary scripts to perform fault simulation and analysis.

- **fault simulation:** We incorporate a mixed-signal simulation at three-level of abstraction, namely, device level simulation for fault model, circuit level simulation for fault site or target module and logic level simulation for rest of the design (not near the fault site). This architecture allows an accurate simulation of fault near the fault site. When the fault is converted to error, we have still an accurate simulation in logic level for the rest of the design.

  Our flow consists of mixed-signal co-simulation of VHDL or Verilog with SPICE. Currently most of industrial CAD simulators support this kind of mixed-signal simulation through Verilog-PLI or other proprietary interface. By default we use a combination of Mentor-Graphics ModelSim for logic simulation and Synopsys HSIM to perform SPICE simulation.

- **evaluation:** After fault co-simulation, simulators produce two traces: i) golden traces, that is free from any fault injection and ii) faulty trace that contains the fault effects. At this step, we investigate the difference between our golden run and faulty runs and report the diferences. From this we extract reliability parameters.

5. DEMONSTRATION

To illustrate how the mechanism of our fault injection flow works, we setup an experimental environment and performed fault injection on selected testcases. The examples chosen for this purpose are 8-bit digital up-down counter (*Counter*), a traffic light controller finite state machine (*FSM*), a 16-bit standard ALU module (*ALU*), a 256-bit 6-Transistor SRAM in spice (*SRAM*) and universal asynchronous receiver/transmitter circuit (*UART*). The library we used for synthesize and simulation is based upon TSMC 0.25um technology. Fault model characteristic and parameters used in this demonstration are described in Table 1. All of them are injected at random time during the simulation. Figure 3 shows an examples of these injections.

Figure 4 contains the result of fault injection. For each testcase we report the failure rate of the circuit. Failure is defined as any deviation from standard (golden) execution in circuit's primary outputs. In many cases, we may inject a fault in circuit however, due to error masking factors, system may not fail and continue to function properly. Failure rate is the total number of failed experiment per total executed experiments (in our case 1000 fault injection per testcase).

6. CONCLUSION

Fault injection is a valuable asset for evaluating circuit's dependability. In this paper we proposed a mixed-signal fault injection flow. We use spice simulation near the fault site and logic co-simulation elsewhere, therefore our flow ensures both accuracy and performance. Our flow incorporates behavioral fault modeling at circuit and device level (in Verilog-A language). By supporting behavioral fault modeling in Verilog-A, researcher can rapidly develop accurate fault models and simulate them on practical designs. Furthermore, it allows users to use their default testbenches and verification scripts, Hence it provide a practical facility to perform fault injection studies. To validate this flow, we have developed different classes of fault models and applied fault injection process on selected circuits.

Fault Model	parameters
SET	Q=10pc, with $TO = TB = 10ns$, Random injection, Two exponential model
EMI	$100MHz$ CW RF signal, Vpeak=$0.5V$, 100ns pulse envlope, Random injection
PLD	$100ns$ duration, Voltage shortage (from $2.5V$ to $0V$) on VDD line, Random injection

Table 1: Fault model details

7. REFERENCES

[1] Mei-Chen Hsueh, Timothy K. Tsai, and Ravishankar K. Iyer. Fault injection techniques and tools. *Computer*, 30:75–82, 1997.

[2] Jeffrey A. Clark and Dhiraj K. Pradhan. Fault injection. *Computer*, 28:47–56, 1995.

[3] Paolo Prinetto Alfredo Benso (Editor). *Fault Injection Techniques and Tools for Embedded Systems Reliability Evaluationn*. Springer US, 2003.

[4] J. Carreira, Henrique Madeira, and J. Gabriel Silva. Xception: A technique for the experimental evaluation of dependability in modern computers. *IEEE Transactions on Software Engineering*, 24:125–136, 1998.

[5] J. H. Barton, E. W. Czeck, Z. Z. Segall, and D. P. Siewiorek. Fault injection experiments using fiat. *IEEE Trans. Comput.*, 39(4):575–582, 1990.

[6] M. Leu K. Echtle. The efa fault injector for fault tolerant distributed system testing. In *Proc. Workshop on Fault-Tolerant Parallel and Distributed Systems*, pages 28–35, Amherst, USA, 1002.

[7] Manuel Rodriguez, Arnaud Albinet, and Jean Arlat. Mafalda-rt: A tool for dependability assessment of real-time systems. *Dependable Systems and Networks, International Conference on*, 0:267, 2002.

[8] H.R. Zarandi, S.G. Miremadi, and A. Ejlali. Dependability analysis using a fault injection tool based on synthesizability of hdl models. In *Defect and Fault Tolerance in VLSI Systems, 2003. Proceedings. 18th IEEE International Symposium on*, pages 485 – 492, 3-5 2003.

[9] J. Boue;, P. Petillon, and Y. Crouzet. Mefisto-l: A vhdl-based fault injection tool for the experimental assessment of fault tolerance. *Fault-Tolerant Computing, International Symposium on*, 0:168, 1998.

[10] E. Jenn, J. Arlat, M. Rimen, J. Ohlsson, and J. Karlsson. Fault injection into vhdl models: the mefisto tool. In *Fault-Tolerant Computing, 1994. FTCS-24. Digest of Papers., Twenty-Fourth International Symposium on*, pages 66 –75, 15-17 1994.

[11] Volkmar Sieh, Oliver Tschche, and Frank Balbach. Verify: Evaluation of reliability using vhdl-models with embedded fault descriptions. *Fault-Tolerant Computing, International Symposium on*, 0:32, 1997.

[12] Kumar K. Goswami, Ravishankar K. Iyer, and Luke Young. Depend: A simulation-based environment for system level dependability analysis. *IEEE Transactions on Computers*, 46:60–74, 1997.

[13] J. Aidemark, J. Vinter, P. Folkesson, and J. Karlsson. Goofi: generic object-oriented fault injection tool. In *Dependable Systems and Networks, 2001. DSN 2001. International Conference on*, pages 83 –88, 1-4 2001.

[14] Henrique Madeira, Mario Zenha Rela, Francisco Moreira, and Joao Gabriel Silva. Rifle: A general purpose pin-level fault injector. In *EDCC-1: Proceedings of the First European Dependable Computing Conference on Dependable*

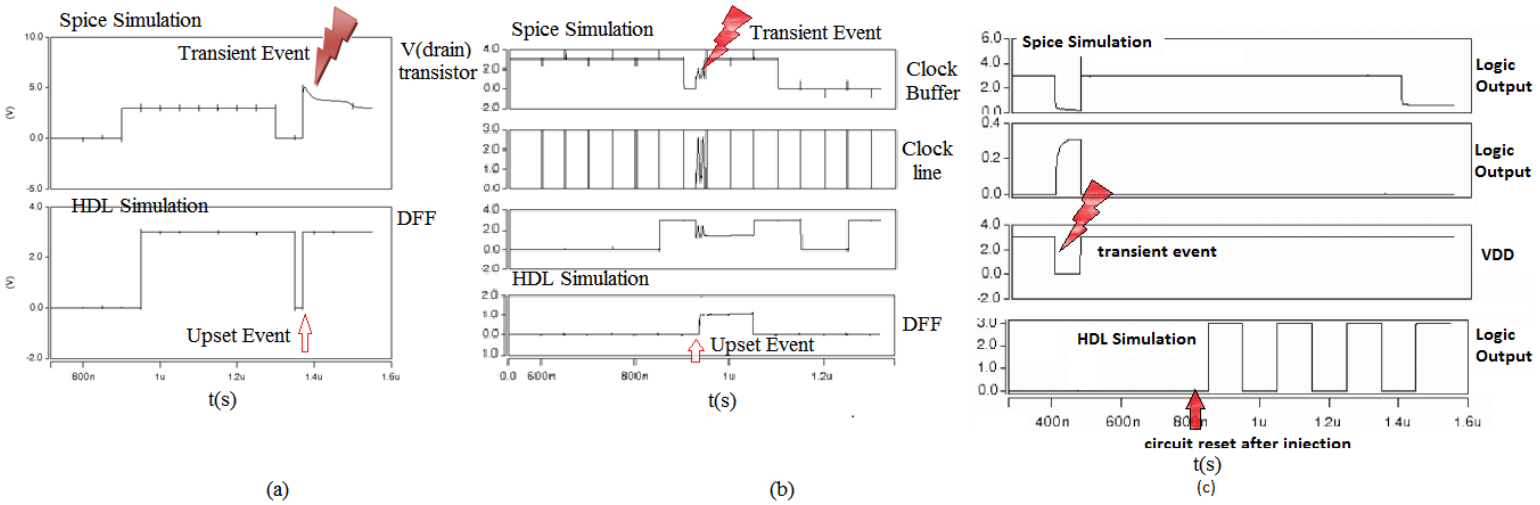

Figure 3: (a) a single event transient injected at circuit level and it's effect at logic level and (b) an example of electro-magnetic pulse injection and (c) a power supply disturbance, power shortage on power network

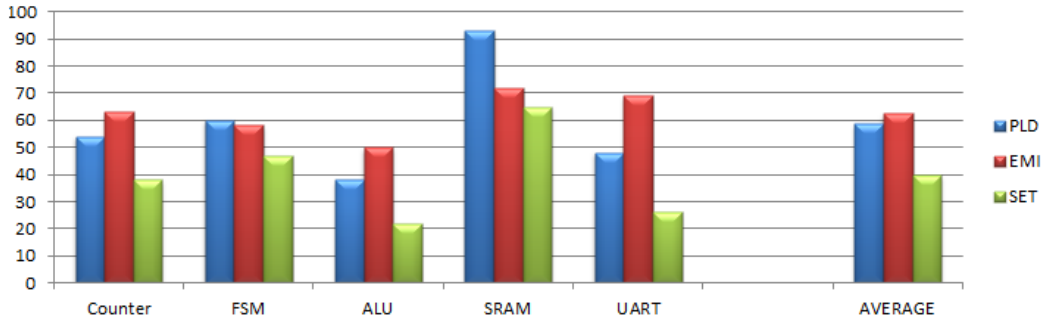

Figure 4: Injection result (System Failure %)

Computing, pages 199–216, London, UK, 1994. Springer-Verlag.

[15] Jean Arlat, Martine Aguera, Louis Amat, Yves Crouzet, Jean-Charles Fabre, Jean-Claude Laprie, Eliane Martins, and David Powell. Fault injection for dependability validation: A methodology and some applications. *IEEE Trans. Softw. Eng.*, 16(2):166–182, 1990.

[16] J. Karlsson and Ulf Gunneflo. Use of heavy-ion radiation from californium-252 for fault injection experiments. *1st IFIP International Working Conference on Dependable Computing for Critical Applications (DCCA-1)*, August 23-25, 1989.

[17] Johan Karlsson, Peter Liden, Peter Dahlgren, Rolf Johansson, and Ulf Gunneflo. Using heavy-ion radiation to validate fault-handling mechanisms. *IEEE Micro*, 14(1):8–11, 13–23, 1994.

[18] G.S. Choi, R.K. Iyer, and V.A. Carreno. Simulated fault injection: a methodology to evaluate fault tolerant microprocessor architectures. *Reliability, IEEE Transactions on*, 39(4):486 –491, oct 1990.

[19] Jim Holmes Matt Francis, Marek Turowski and Alan Mantooth. Efficient modeling of single event transients directly in compact device models. *BMAS*, 2007.

[20] R. Leveugle and A. Ammari. Early seu fault injection in digital, analog and mixed signal circuits: A global flow. In *DATE '04: Proceedings of the conference on Design, automation and test in Europe*, page 10590, Washington, DC, USA, 2004. IEEE Computer Society.

[21] Peter R. Wilson, Yavuz Kiliç, J. Neil Ross, Mark Zwolinski, and Andrew D. Brown. Behavioural modelling of operational amplifier faults using vhdl-ams. *IEEE Transactions on Neural Networks*, 4:740–747, 2002.

[22] A. Ammari, L. Anghel, and R. Leveugle. Set fault injection methods in analog circuits: Case study.

[23] Tino Heijmen and Author(s) Tino Heijmen. Radiation-induced soft errors in digital circuits - a literature survey, 2002.

[24] L.B. Freeman. Critical charge calculations for a bipolar SRAM array. *IBM Journal of Research and Development*, 40(1):119–129, January 1996.

[25] T. Merelle, H. Chabane, J.-M. Palau, K. Castellani-Coulie, F. Wrobel, F. Saigne, B. Sagnes, J. Boch, J. R. Vaille, G. Gasiot, P. Roche, M.-C. Palau, and T. Carriere. Criterion for SEU Occurrence in SRAM Deduced From Circuit and Device Simulations in Case of Neutron-Induced SER. *IEEE Transactions on Nuclear Science*, 52:1148–1155, August 2005.

[26] G. Gaidano V. Pozzolo F. Fiori, S. Benelli. Investigation on vlsis input ports susceptibility to conducted rf interference. *IEEE International Symposium on Electromagnetic Compatibility*, 1997.

[27] F. Fiori. Integrated circuit susceptibility to conducted rf interference. *Compliance Engineering*, 17(8):40–49, 2000.

[28] F. Fiori. A new nonlinear model of emi-induced distortion phenomena in. cmos operational amplifiers. *IEEE Trans. on Electr. Comp.*, 2002.

[29] Howard H. Chen, David D. Ling, Deep submicron Vlsi, and Chip Design. Power supply noise analysis methodology for deep-submicron vlsi chip design, 1997.

[30] Shiyou Zhao and Kaushik Roy. Estimation of switching noise on power supply lines in deep sub-micron cmos circuits. *VLSI Design, International Conference on*, 0:168, 2000.

[31] Yi-Shing Chang, Sandeep K. Gupta, and Melvin A. Breuer. Analysis of ground bounce in deep sub-micron circuits. *VLSI Test Symposium, IEEE*, 0:110, 1997.