Laser-Induced Fault Simulation

Feng Lu, Giorgio Di Natale, Marie-Lise Flottes, Bruno Rouzeyre

2013 16th Euromicro Conference on Digital System Design

Laser-Induced Fault Simulation

Feng Lu, Giorgio Di Natale, Marie-Lise Flottes, Bruno Rouzeyre
LIRMM (Université Montpellier II /CNRS UMR 5506)
Montpellier, France
{lu, dinatale, flottes, rouzeyre}@lirmm.fr

***Abstract*—This paper presents a multi-level simulator for laser-induced fault simulation in digital circuits. It automatically performs the simulation of laser-induced faults using layout information and laser spot information in order to locate affected gates and derive fault-models. The paper mainly focuses on multi-level simulation for obtaining high accuracy of the fault simulation at transistor level and high speed for the simulation of the rest of the circuit. This multi-level process allows handling natural and maliciously induced physical phenomenon leading to circuit misbehavior, while dealing with large circuits.**

Keywords – Multi-level Fault Simulation, Layout-Oriented Fault Simulation, PLS Effects

I. INTRODUCTION

The Photoelectric Laser Stimulation (PLS) is a technique that allows injecting photoelectric energy by pulsed laser into the substrate of an electronic device. The injected energy produces photoelectron-hole pairs that can result in a transient fault of the affected logic gates. This method was first used for the simulation of Single Event Effect (SEE) in 1965 [1], because the nature of this fault is similar to the ionization effect generated by energetic particles.

With the increase in integration densities, quality and dependability of integrated circuits have become strong requirements for digital device. PLS technique is now becoming an effective method for fault analysis. Moreover, several modern hardware devices (such as cellular phones, e-tablets, credit cards) require security and privacy protection. For achieving the high security level, secure protocols and strong encryption algorithms are widely studied. However, the hardware that implements the secure algorithms and protocols is becoming the focus of attacks [2]. Among all types of attacks performed on the hardware part of the system, fault attacks have proven to be very effective. By provoking an error during an encryption process via laser attacks, the secret key may be retrieved.

While PLS technique is very effective, it has anyway several limitations. Indeed, besides the cost of the hardware platform, it requires the device to be already manufactured. Unfortunately, this is not possible when evaluating at design-time the effectiveness of counter-measures against fault attacks. Fault simulation is therefore the widely adopted solution to analyze the behavior of a device under the effect of PLS. Nevertheless, fault simulation is a time consuming process. Its complexity depends on the accuracy in the description of the physical perturbation. For instance, faults due to PLS are often modeled by bit-flips in the storing elements of the circuit, and statistical approaches are used to estimate the rate of this error.

Unfortunately, these models are showing their inefficiency due to the shrinking size of transistors. For instance, radiations and laser attacks can affect multiple gates at the same time. The relation of the fault effect on multiple gates cannot be easily described with classic fault models. Conversely, electrical and physical simulations would permit very high accuracy. However, the execution time required for this fine-grained simulation would not be acceptable for today's large circuits.

Nowadays there is a lack of simulation tools able to take into account at the same time a local perturbation that requires fine-grained simulation and the scalability to the whole circuit. In this paper we present a fault simulator able to perform, besides classical fault models, multi-level fault simulations of circuit's perturbations described at electrical level. Our tool automatically takes into account the layout of the circuit and the laser's parameters (energy, size of the spot, position, duration) to simulate the effect of PLS. The main idea behind the multi-level simulation is to run a quick simulation of the circuit to reach the moment when the perturbation has an effect on the circuit. Then, the accuracy of the simulation is increased to clearly identify the input waveforms for the affected sub-circuit, which is electrically simulated with and without the effect of the perturbation. The differences between faulty and fault-free simulations are abstracted at higher level to terminate the simulation of the whole circuit at gate level. We will show how this tool improves the execution time for large circuits, while keeping high accuracy of experimental results.

The paper is organized as follows: Section 2 provides background and state-of-the-art. Section 3 describes the architecture of the fault simulator, whereas Section 4 elaborates the proposed multi-level laser induced fault simulation method. Section 5 presents some experimental simulation results. Eventually, Section 6 concludes the paper and draws future perspectives of this work.

II. STATE-OF-THE-ART

Fault simulations can be performed at different abstraction levels (physical, transistor, gate, RTL, system). Low abstraction levels provide high accuracy in terms of simulation results, while higher levels allow simulating bigger systems in reasonable time. Because of the increase in the transistor density and the clock frequency, ionizing effects may affect multiple gates at the same time and for period of times longer than one clock cycle. These effects must be simulated at low abstraction levels to be representative of the physical phenomenon. However the possible consequences of these effects must be analyzed and propagated to the whole system. Therefore, multi-level fault simulators become a need for this type of phenomena. We present in the next paragraph a brief summary of existing techniques and tools for multi-level fault simulation.

Paper [3] presents a gate-level simulation environment for alpha-particle-induced transient faults. It includes two simulation engines for both annotated-delay and zero-delay simulations. For a pulse of given width, flip-flops have a latching window such that if the pulse arrives inside the window, it will be latched, and if it arrives outside the

`/13 $26.00 © 2013 IEEE
DOI 10.1109/DSD.2013.72

window, it will be ignored by the flip-flop. Author defined a set of "latching windows" corresponding to the different widths of transient pulse for each flip-flop of the standard cell. The timing logic-level simulation is operated with the arrival time of the transient pulse and the "latching windows ", if the arrival time of pulse falls in the range of its latching window the latch will be flipped, and on the contrary the latch maintains the correct value. It has been shown that an improvement in simulation time is achieved by using the fault-driven algorithm as opposed to the standard event-driven algorithm to perform fault injection. Once the transient faults have been latched, the timing simulator is no longer needed and further speedup is achieved by using a zero-delay parallel fault simulator. Unfortunately, as a gate-level fault simulator, the fault model is defined as a logic pulse with different widths for the analogic phenomenon of α-particle injection. Moreover, for their experiments, charges are injected only at the output nodes of gates. All the constraints of simulation will not hold in reality, it represents a reasonable compromise between modeling effort and usefulness of fault simulation results.

A multilevel fault simulator operating at switch- and gate-levels is described in [4]. It supports the inertial delay model. In this approach, low/high resistances are used to model on/off states of transistors. Its major advantage is the accuracy with respect to both fault-models and timing. Classical switch-level simulation, transistor conductance and node capacitance are modeled by discrete strengths. But by assuming strengths for transistors, switch-level simulation is not sufficient to simulate the fault caused by ionizing effects.

The work in [5] uses event-based mixed-level fault simulation to simulate the effect of manufacturing defects more accurately while maintaining a tolerable simulation time. It presents a set of fault models for CMOS realistic bridging (BRI) and Line-Open (LOP) faults for efficient fault simulation. But it does not consider transistor-level and therefore sacrificed accuracy. In addition, the variety of the fault models considered was quite limited for the simulation of different faults injection.

A multilevel fault-simulator for bridging faults is expounded in [6]. It supports the gate-, switch- and electrical-level fault simulation. A bridging fault defines a region to which SPICE-like simulation is applied, trading-off speed and accuracy. The region is defined by the actual fault sites plus some digital levels forward in the circuit. The region is large enough to assure that only power-supply rail voltages are passed into the gate-level portions of the circuit. Unfortunately, the implementation is insensitive to timing; large memory required and limited performance caused by the electrical simulation.

III. LOGIC SIMULATOR ARCHITECTURE

tLIFTING is fault simulator based on the open-source Lifting [7]. tLIFTING allows both 0-delay and delay-annotated simulations of digital circuits described in Verilog, as sketched in Fig. 1.

The fault simulator reads the netlist of the circuit described in verilog (.v), and the input test sequence described in a proprietary format (.ts).

Moreover, the simulator can read the Delay-Annotated file, which provides information related to the delays of each gate in the Standard Delay Format (.sdf), and the fault list, which explicitly define the faults to take into account.

In order to handle any technological library, the simulator integrates a converter that, starting from the information of each cell (I/O number, pin names, truth table, and input capacitance), generates a corresponding tLifting-compatible description (i.e., in C++).

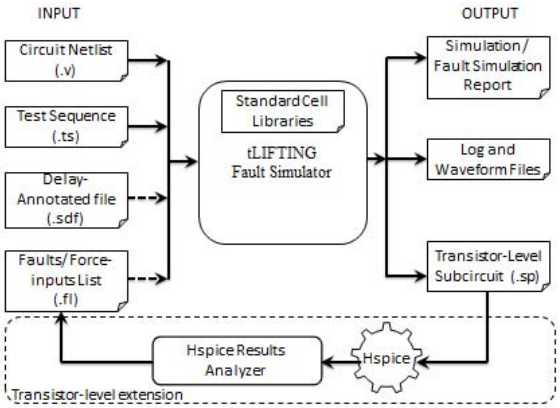

Figure 1. tLIFTING architecture

A. *delay logic simulation*

The basic idea of the simulator is that each circuit gate is modeled as a C++ class. The root of the object model hierarchy is the abstract class *generic_gate*. It includes the information of a common digital gate such as its name, number and current values of inputs and outputs, and the list of gates connected to each output. Moreover, a method is implemented to set the value of an input (*set_input*) while a virtual method (*calculate_output*) is used to define the logic function of the gate. Each standard cell is written as a class that inherits from *generic_gate* and that implements this method in order to specify the functionality of the gate.

When the simulator reads the netlist of the circuit, *generic_gates* are instantiated and linked one to the others by building the proper *fan_out* array. This array stores, for each *generic_gate*'s output port, all the input ports of the gates connected to it.

The main method of *generic_gate* is *set_input*. It resorts to the *calculate_output* method to determine the new output value of the gate. When the new value is different from the old one, it propagates the new value to each gate connected at the output by invoking the *set_input* method of those gates. This recursive method allows propagating any signal variation up to primary outputs of the circuit.

B. *Timing Simulation*

tLIFTING implements an event-driven simulator engine to allow delay-annotated simulations. It is based on a priority queue to store simulation events. Each event is characterized by the time and the type. The queue is sorted according to the event time, and a new element inserted in the queue will be placed accordingly. While the priority queue is not empty, the simulator pops the first event and executes the corresponding action according to type of event.

The simulator manages 4 types of event: Primary Input (PI) event, Propagation event, Fault Injection (FI) event and Fault Release (FR) event. PI events are generated when reading the input stimuli file. This event means that a primary input value is presenting. Propagation events are created whenever the output of a gate changes its value. The corresponding event time is set to the current time plus

the propagation delay of the gate. FI and FR events are used to handle faulty behaviors. FI event set the value of a specific port to 0 or 1 during the simulation, no matter the actual value of the signal. On the contrary, FR event removes the previously applied FI fault and restores the proper value of the specific port.

The delay information, which is given by design tools, can be more or less precise based on the design level. For instance, before placement and routing there is not the delay of wires while the delay between the instant the gate input switches and the instant the gate output switches is included.

Before starting the simulation, all events related to faults are created according to what defined in the fault list. These events are then inserted in the priority queue.

IV. MULTI-LEVEL PLS FAULT SIMULATION

tLIFTING enables multi-level fault simulation to analyze the effect of transient faults generated by PLS effects, which can be described only at electrical level. It automatically performs the simulation of the whole circuit at gate level before the fault appearance, then the fault simulation at electrical level of only the gates involved in the fault, to move again to the gate level to finish the simulation. It improves the simulation run time compared to full transistor-level fault simulation.

A. Electrical Fault Model

A transient fault is not the result of a circuit defect but results from an external phenomenon such as a laser pulse. When a beam of laser passes through a micro-electronic device, it will have an ionizing effect that triggers the formation of a dense track of electron-hole pairs. The separation and diffusion of these electron-hole pairs can produce a current transient which is subsequently observed on the signal node as a transient fault pulse. Normally, the transient fault is not considered permanently damaging to the transistor's or circuits' functionality.

When a transistor is illuminated by a laser pulse, a fault may be observed or not at the output of the gate depending on many physical parameters such as the laser intensity, wavelength, or load capacitance of the illuminated gate. Thus, a logic-level model of the circuit is inadequate to properly simulate these effects. Once an electrical model of the phenomenon is built, one can rely on an analog simulator such as Spice to check whether a fault pulse is generated on the gate output or not. If so, the fault pulse parameters (time, duration) can be also determined. Several models have been proposed in the literature for laser-inducted effect (e.g., [8], [9]). The basic idea of these models is that the laser affect can be modeled by a current source.

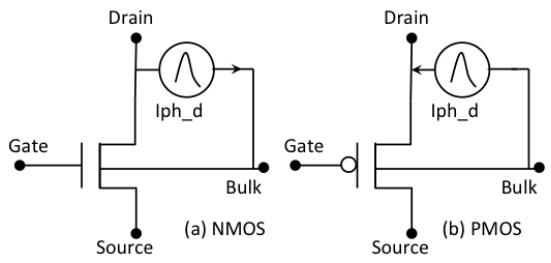

Figure 2. The simple electrical model for a fault induced by laser injection

In this paper, we use a simple model depicted in Fig. 2. Nevertheless, more sophisticated models can be used instead since the model is a parameter of the fault simulator.

B. Multi-Level Fault Simulation Method

Unfortunately, electrical simulation is extremely CPU intensive and can be performed only on a small portion of a circuit in reasonable time.

To circumvent the drawbacks of both approaches, we propose a method for fault simulation of a digital circuit affected by a perturbation described at electrical level. The term multi-level means that the fault is injected into the transistor-level description of the affected gate(s), while other gates are simulated at higher abstraction levels. The justification of the multi-level simulation resides in the fact that the fault model cannot be abstracted at a logic level but requires a dedicated simulation at electrical level. On the other hand, the whole circuit does not necessitate a complete simulation at electrical level since most part of the circuit is not affected and behaves as in the fault-free scenario. This multi-level fault simulation flow thus intends to improve simulation run time compared to full transistor-level fault simulation.

Fig. 3 shows the topological partitioning of the circuit for the multi-level simulation. The gates directly affected by the laser spot are shown in the gates $\mathbb{G}$. We shall define sub-circuit $\mathbb{C}$ as the set of gates in the input and output cones of $\mathbb{G}$. Finally, $\mathbb{S}$ is the whole system.

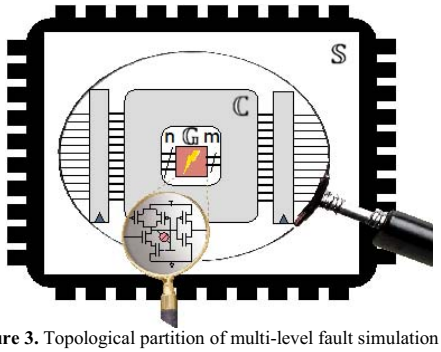

Figure 3. Topological partition of multi-level fault simulation

The first step of laser induced fault simulation is to localize the illuminated gates $\mathbb{G}$ from the circuit layout description and from the laser position and its spot size. At the same time the electrical fault model parameters are calculated via the laser's physical information such as laser's intensity and wavelength. Then, the sub-circuit $\mathbb{C}$ is identified. It contains, besides $\mathbb{G}$, all gates in $\mathbb{G}$'s input and output cones, limited to primary I/Os or sequential elements.

C. Localization affected component

For abstracting circuit layout's topological information, we used LEF (Library Exchange Format) and DEF (Design Exchange Format) files. The first defines the geometry (size and shape) of each element of the technological library, while the second defines the position of each gate within the circuit, including the netlist and design constraints (see Fig. 4 for more details).

Therefore, starting from the laser's parameters (in particular, the position and the size of the spot), it is possible to extract with are the gates illuminated by the laser.

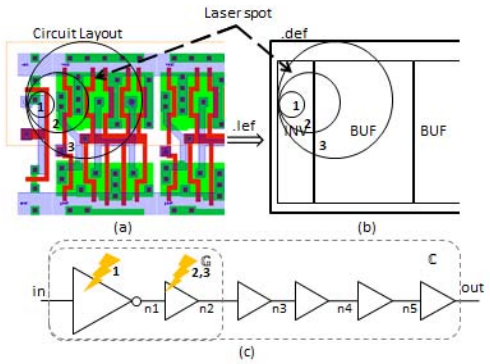

Figure 4. Abstract the illuminated sub-circuit from circuit layout to logic-level netlist. (a) Transistors illuminated by 3 different size laser spots; (b) Corresponding gates; (c) Sub-circuit with the perturbed gates

To precisely locate which are the affected transistors within a standard cell, we use the GDSII file of the technological library. Starting from these data, we can therefore zoom to the affected transistors (Fig. 5 shows the perturbed transistors which correspond to the layout of the Fig. 4.a).

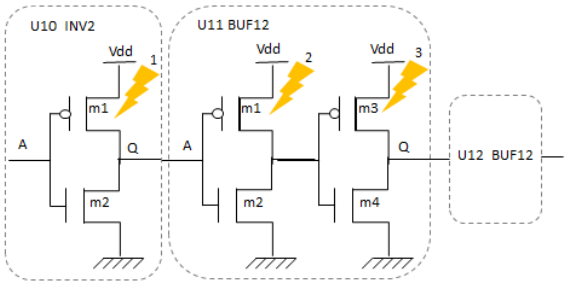

Figure 5. Locating the affected transistor(s) in the illuminated sub-circuit

D. *Laser-Induced Fault List*

Faults caused by ionizing effects can be simulated by specifying the fault list in terms of laser's parameters and position of the injection. Starting from this list, the faults are modeled and injected into the transistor-level circuit element description for the transistor-level fault simulation. For instance, the example shown in Fig. 2 can be described with the following syntax:

```
ionize cmos_name  current_value
```

The fault is injected between the drain and the bulk of the `cmos_name` transistor according to Section IV.A. The current source modeling the fault is a double exponential current source (Fig. 6) with the highest current value equal to `current_value`.

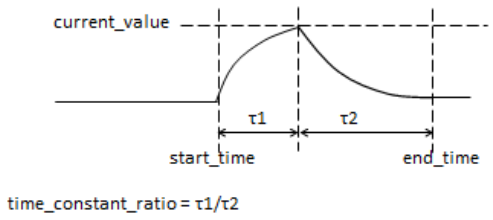

Figure 6. The schematic diagram of exponential source

The parameters of start and end time of the fault pulse and the ratio of rise/fall time constant are defined by additional key word `time` for each faulty transistor in the fault list:

```
time start_time  end_time  time_constant_ratio
```

The following example represents the complete laser fault list for the example shown in Fig.5 in which the laser spot 3 affects 3 PMOS transistors belonging to the inverter U10 and the buffer U11.

```
gate U10 INV2 {
      ionize m1 0.40mA
      time 19ns 21ns 50%
}
gate U11 BUF12 {
      ionize m1 1.20mA
      time 19ns 21ns 50%
      ionize m3 4.80mA
      time 19ns 21ns 50%
}
gate U12 BUF12 {}
```

In the fault list, users can also define fault-free gates in order to force the simulator to include them in the transistor-level simulation.

E. *Process of multi-level fault simulation*

Fig. 7 shows the timing partitioning, which will be used for the overall simulation process. The simulation starts at T_0. The laser perturbation affecting $\mathbb{G}$ lasts from T_{Pon} to T_{Poff}. To accurately take into account the effects of the perturbation on the whole circuit, we decided to consider the clock limits around the perturbation for the fine-grained simulation (instants T_{Start} and T_{Stop}).

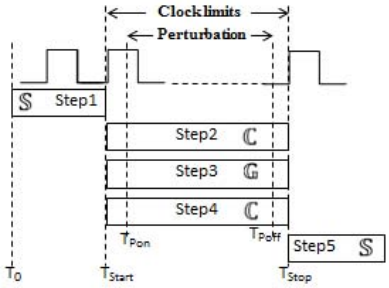

Figure 7. Timing partitioning and steps

The proposed multi-level fault simulation involves 5 steps during which the different parts of the system are simulated at different time periods, abstraction levels and accuracy as follows:

1. The whole system $\mathbb{S}$ is simulated from T_0 to T_{Start} in order to compute the state of $\mathbb{C}$ just before fault injection.
2. The sub-circuit $\mathbb{C}$ is simulated from T_{Start} to T_{Stop} by taking into account the delay of each gate and nets. This simulation can be more or less accurate based on the precision of the delays. The goal of this step is to extract the waveforms for the input and output bits of $\mathbb{G}$ during the perturbation.
3. $\mathbb{G}$ is modified to include the electrical model of the perturbation. It is then simulated at electrical level by using the input waveforms obtained in step 2. The output analog waveform is translated to logical levels and compared with the nominal logic values obtained in step 2 to create a list of timed logic faults.
4. $\mathbb{C}$ is fault simulated from T_{Start} to T_{Stop} using the faults defined in step 3 and taking into account the delay of

nets and gates. The goal is to compute the state of ℂ. If the state is equal to the one obtained in Step 2, the perturbation has not effect on the circuit and the simulation is stopped.

5. If, on the contrary, the states are different, the whole system 𝕊 is simulated starting from the faulty state up to the end of the simulation at logic level.

The combinational circuit which we have considered as the ℂ sub-circuit given in Fig. 4.c is used to illustrate the step 2 to step 4 of this process. We shall assume that the pulse laser spot 1 illumines a certain parts of circuit's layout (Fig. 4.a). From the abstracted layout information, the perturbed gates 𝔾 are defined (Fig 4.b). At first, circuit ℂ is timed-simulated taking into account the delays of the gates (delays extracted from the foundry's library). During this period, the logical waveforms at points "in" and "n2" are recorded. Let's assume that waveform at "n2" is the one depicted in Fig. 8.a.

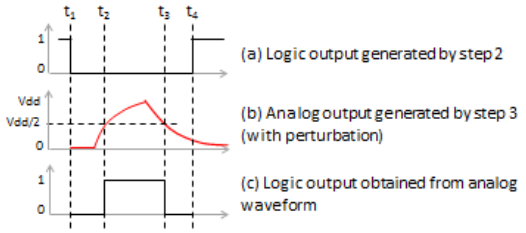

Figure 8. Simulation step 3

Using the waveform at point "in", 𝔾 is electrically simulated leading to the waveforms given in Fig. 8.b. According to the threshold voltage, this waveform is translated into the logic domain to the one given in Fig. 8.c. From the difference between waveforms (a) and (c), it is inferred that an SET occurs on 𝔾's output "n2" from t_2 to t_3. Then, ℂ is fault-simulated with this fault and timing information.

V. Experimental Results

Our fault simulator implements 2 main simulator engines: the first for 0-delay logic simulations (fast recursive algorithm) and the second for delay-annotated simulations (event-driven algorithm). Moreover, it implements the PLS fault multi-level simulation flow described as follows:

- Steps 1 and 5 are executed with the 0-delay engine;
- Steps 2 is executed with delay-annotated engine;
- Step 3 is executed by automatically generating the source files and scripts for the Hspice simulator. The output of Hspice is then elaborated to generate the fault list for the higher level;
- Step 4 is executed with delay-annotated engine by taking into account the faults generated in step 3.

The two next sub-sections compare the execution time and the accuracy of the results.

A. Execution Time of Multi-Level Fault Simulation

In this sub-section we show the performances of the proposed method. We compare the execution time of the multi-level simulation with respect to pure Hspice simulation. Since performing steps 1 and 5 with Hspice simulation would be both very long and without any interest, we conducted our experiments on combinational circuits for which steps 1 and 5 are not needed.

We have set up a fault simulation experiment for each circuit taken from the combinational *ISCAS85* benchmarks, which we have considered as the ℂ sub-circuit. We injected a current pulse lasting 2ns within a PMOS transistor of an inverter in the circuit, as the one shown in Fig. 2. For each benchmark we properly selected the inverter and the energy of the pulse such that it modifies the final state of the circuit.

Table 1. Execution time comparison

			Fault simulation [s]		
circuit	Size	Vectors	Multi-Level	Hspice	Factor
c432	112	77	0.116	55.78	480x
c499	133	77	0.132	89.05	674x
c1355	162	72	0.132	86.48	655x
c1908	169	80	0.228	129.12	566x
c880	204	77	0.404	119.22	295x
c2670	292	136	2.052	393.78	191x
c3540	476	171	1.916	676.7	353x
c5315	608	113	4.204	705.05	167x
c7552	705	167	9.428	1356.56	143x
c6288	1286	63	8.308	1248.75	150x
b01	31	13	0.03	6	200x
b04	41	16	0.04	5.03	125x
b10	89	128	0.09	25.97	288x

Table 1 summarizes the experimental results. For each benchmark, we reported its size in equivalent gates and the number of simulated vectors. The next two columns compare the execution times of the fault simulation between the multi-level approach and the full Hspice simulation. The increase in speed is given in the last column. It ranges from 100x for the smallest circuit, up to more than 650x for the biggest benchmark. Since the electrical level simulation is limited to a very small part of the circuit, the larger the circuit, the higher the benefit of multi-level simulation.

B. Precision Analysis of Fault Simulation

In this sub-section we evaluate the accuracy of the multi-level simulation compared to the result obtained from Hspice. We first define a metric to measure the error between the two simulations. We considered the following (see Fig. 9):

- Signals generated by Hspice simulation are translated to logic levels. Thereinafter, the terms "generated by Hspice" refers to logic signals translated from the analog ones;
- te_i is a time interval during which the output generated by Hspice simulation is different from the one generated by tLIFTING;
- The total error time is defined as $T_e = \sum te_i$
- T_f is defined as the smallest interval that contains all faults produced by either simulation.

Finally, the error between multi-level and Hspice simulations is calculated as $Accuracy = 1 - T_e/T_f$.

We set up the experiment shown in Fig. 10. The circuit under perturbation is composed of a single inverter, followed by 5 buffers. We have injected a fault in the PMOS transistor of this inverter (as in Fig. 2). The value of

the current source that corresponds to the fault is in the range between 0µA (no fault) to 500µA, by step of 5µA.

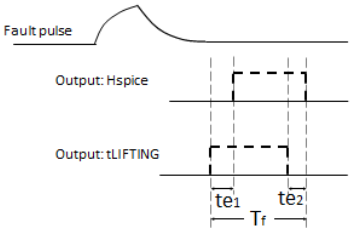

Figure 9. Error calculation

To understand the accuracy loss of the multi-level simulation, in addition to performing the Hspice simulation of the first inverter only, we also considered several partitioning of the circuit. For instance, Level-3 in Fig. 10 means that Hspice simulation is performed for the first inverter and the 2 next buffers, the conversion from analog to digital is done at n3 signal, and the simulation is then terminated at logic level for the remaining gates.

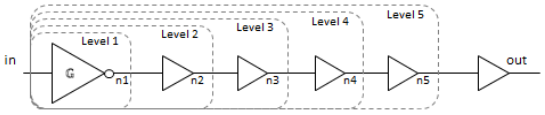

Figure 10. Scheme of the test circuit

When the current pulse is less than 310µA, both simulations do not produce any error at the output of the circuit. For higher values of the pulse, the 6 simulations (the pure Hspice simulation plus the 5 combinations described above) show an error at the output. However, the errors generated by tLIFTING do not exactly match in time and duration the one generated by pure Hspice simulation. These statistical characteristics of discrepancy are given in Table 2 by means of the total error time for the 5 levels of multi-level simulation experiments.

Table 2. Statistical Characteristics of Discrepancy

Range of fault amplitudes (310uA - 500uA)	Level-1	Level-2	Level-3	Level-4	Level-5
Average of Te [ps]	149.27	158.41	123.61	85.07	46.07
Standard deviation of Te	49.97	2.97	2.80	2.95	2.90

The average of T_e can be understood as the shift of the two fault pulses (generated by Hspice and tLIFTING). Clearly, the higher number of logic levels being simulated with Hspice, the lesser discrepancy in time. Nevertheless, from the level-2 the standard deviations of T_e become stable and maintained at a low deviation. It means that after the fault pulse is digitized by a following component (as level-2 shown in Fig. 10) the T_e can be considered as a systematic error which does not change with the width of the fault pulse. This systematic error comes from the accumulation of differences between Hspice and the logic-level SDF description.

As a solution, the transistor-level simulation is proposed to perform at least in level-2 scale. Considering the worst case of all the experiments for level-2 and the above, the narrowest fault pulse which leads to the minimum value of T_f is presented in the level-2 experiment with the current pulse of 310µA (as in Fig.11). In this case we obtained 81.4% of Accuracy, we always generated correct results (i.e., the two simulations show a logic error at the output of the circuit). The Accuracy metric we used in this experiment is very pessimistic because it considers as T_f the smallest interval that contains the discrepancy. However, the discrepancy is only relevant when a different logic value is actually latched in the storing elements. If the difference occurs earlier or later than the clock event, the difference is meaningless.

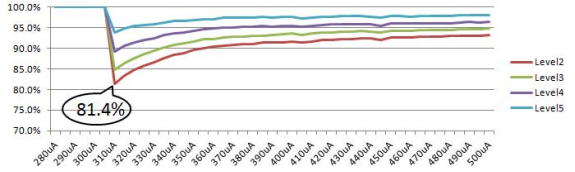

Figure 11. Accuracy vs. current pulse amplitude

VI. CONCLUSIONS

In this paper we presented a multi-level laser induced fault simulation method that enables studying the effect of faults modeled at transistor level to large digital circuits. The proposed approach is based on a combination of electrical-level simulation for the sub-circuit affected by the fault and logic level simulation of the rest of the circuit to speed up the whole process. This method will be extensively exploited in the process of evaluating countermeasures against fault attacks, as well as in the reliability evaluation of deep sub-micron devices.

VII. ACKNOWLEDGEMENTS

This work has been supported by the contracts FUI CALISSON 2 (DGCIS AAP10) and ANR LIESSE (ANR-12-INSE-0008-01).

REFERENCES

[1] Habing, Donald H., “The use of lasers to simulate radiation-induced transients in semiconductor devices and circuits”, IEEE Trans. Nucl. Sci., 1965, 12(5): 91--100.

[2] Kris Tiri, Ingrid Verbauwhede, “Simulation models for side-channel information leaks”, Proceedings of the 42nd annual Design Automation Conference, New York, 2005.

[3] C. Hungse, E.M. Rudnick, J.H. Patel, R.K. Iyer, G.S. Choi, “a gate-level simulation environment for alpha-particle-induced transient faults,” IEEE Transactions on Computers, Vol. 45, No. 11, Nov. 1996, pp. 1248-1256

[4] Meyer W, Camposano R, “Active timing multilevel fault-simulation with switch-level accuracy,” IEEE Trans CAD Integr Circ Syst, 14: 1241–1256, 1995.

[5] Santos M B, Teixeira J P, “Defect-oriented mixed-level fault simulation of digital systems-on-a-chip using HDL,” Proceedings of the Conference on Design, Automation and Test in Europe, Munich, Germany, 1999.

[6] G. S. Greenstein and J. H. Patel, “E-PROOFS: A CMOS bridging fault simulator,” in Proc. Int. Conf. Computer-Aided Design, 1992, pp.268-271.

[7] A. Bosio, G. Di Natale, “LIFTING: A Flexible Open-Source Fault Simulator”, 17th Asian Test Symposium, Sapporo, 2008, pp. 35 - 40.

[8] C. Godlewski, V. Pouget, D. Lewis, M. Lisart, “Electrical modeling of the effect of beam profile for pulsed laser fault injection”, 20th European Symposium on the Reliability of Electron Devices, Failure Physics and Analysis, Volume 49, Issues 9-11, September-November 2009, pp. 1143-1147.

[9] A. Sarafianos, R. Llido, J.M. Dutertre, O. Gagliano, V. Serradeil, M. Lisart, V. Goubier, A. Tria, V. Pouget, D. Lewis, “Building the electrical model of the Photoelectric Laser Stimulation of a PMOS transistor in 90 nm technology”, Microelectronics Reliability, Volume 52, Issues 9–10, September–October 2012, pp. 2035–2038.