Multi-Level Simulated Fault Injection for Data Dependent Reliability Analysis of RTL Circuit Descriptions

Sergiu NIMARA, Alexandru AMARICAI, Oana BONCALO, Mircea POPA
University Politehnica Timisoara
Vasile Parvan Blvd, Nr.2, 300223, Timisoara, Romania
alexandru.amaricai@cs.upt.ro

***Abstract*— This paper proposes data dependent reliability evaluation methodology for digital systems described at Register Transfer Level (RTL). It uses a hybrid hierarchical approach, combining the accuracy provided by Gate Level (GL) Simulated Fault Injection (SFI) and the low simulation overhead required by RTL fault injection. The methodology comprises the following steps: the correct simulation of the RTL system, according to a set of input vectors, hierarchical decomposition of the system into basic RTL blocks, logic synthesis of basic RTL blocks, data dependent SFI for the GL netlists, and RTL SFI. The proposed methodology has been validated in terms of accuracy on a medium sized circuit – the parallel comparator used in Check Node Unit (CNU) of the Low-Density Parity-Check (LDPC) decoders. The methodology has been applied for the reliability analysis of a 128-bit Advanced Encryption Standard (AES) crypto-core, for which the GL simulation was prohibitive in terms of required computational resources.**

***Index Terms*—Digital Circuits, Probabilistic Circuits, Register Transfer Level, Reliability, Simulated Fault Injection**

I. INTRODUCTION

Reliability represents one of the most important issues in todays' deep nanometer CMOS technologies. On one hand, the wide process variations associated to these technologies, coupled with process and temperature variations, lead to timing variations of the basic logic devices. On the other hand, aggressive techniques to tackle the power wall, such the supply voltage scaling till near and sub threshold values, lead to decreased noise margins in digital circuit. In this context, CMOS circuits supplied at near and sub-threshold voltages, display a probabilistic behavior: the probability of a logic gate to have a correct output is less than 1 [1]-[5]. In this context, reliability estimation methodologies are required to correctly assess the impact of aggressive voltage scaling in early design stages of deep nanometer CMOS circuits.

Reliability analysis can be performed using analytical methods, simulations or prototype based analysis [5]. Simulation based methods provide good trade-off between the accuracy of the results and the costs of the evaluation. They rely on simulated fault injection (SFI) [6][8]. There are two important factors regarding SFI: fault modeling capability and simulation time. Good fault modeling capability is obtained when simulation is performed at low level of abstractions (such as transistor level simulations in SPICE). However, low level simulations are computationally intensive; their simulation time is very high and they have high computational requirements (memory and processing). Simulating systems at higher abstraction layers is advantageous from a simulation overhead perspective; however, fault modeling capability is low.

Regarding the probabilistic fault analysis, in [9] it has been indicated that one of the most important factors of the probabilistic behavior of sub-powered logic gates is represented by the inability of the gate to correctly switch in a given amount of time [10]. Furthermore, the error probability has shown a strong data dependency, being influenced by the input switch combinations. In [11-12], GL SFI for data dependent probabilistic fault analysis has been developed. However, GL simulation is unfeasible for complex digital systems.

This paper proposes a two level simulation methodology, which aims at combining the fault modeling capability of obtained at GL, with the simulation time of the RTL SFI. Regarding the GL simulations, they aim at capturing in an accurate way the data dependency characteristic to the probabilistic sub-powered CMOS logic devices. The GL simulations are performed for smaller blocks. The inputs for GL simulations have been extracted based on the correct RTL simulation. The error probabilities obtained after data dependent GL SFI are used to derive the probabilistic saboteurs for the RTL SFI. The latter is performed in order to estimate the whole circuit reliability.

This paper is organized as follows: Section II is dedicated to the related work and other multi-level SFI approaches; the data dependent GL SFI is presented in Section III; the proposed approach is presented in Section IV; Section V is dedicated to the two case studies.

II. SIMULATED FAULT INJECTION FOR RTL CIRCUITS DESCRIPTIONS

SFI has been widely used for reliability assessment of digital systems affected by different types of faults in early design phases [6][8][13]. Regarding the SFI techniques for circuits modeled using Hardware Description Languages (HDL), they are classified in two categories:

1. *Techniques based on simulator commands* – These techniques use commercial HDL simulator commands to introduce faults into the HDL design. Their advantage is represented by the fact that they do not require

This work has been supported by the European Commision Framework Program 7, project "i-Risc: Innovative Reliable Chip Design from Low Power Unreliable Components", under Grant Agreement 309129

modification of the HDL code. Their disadvantage is represented by the fact that their fault modeling capability is limited and highly dependent on the simulator capability

2. *Techniques based on HDL code modifications* – These techniques have the advantage of a high fault modeling capability. These can be further classified in two sub-categories:
   2.1. *Mutants* – Entities/modules which replace the correct description of a component
   2.2. *Saboteurs* – Entities/modules which alter the value and timing characteristics of a signal

SFI performed for RTL circuit descriptions have been used for analysis of faults affecting the digital systems. A wide range of works concentrating on developing SFI components for RTL descriptions or improving the simulation overhead of RTL based analysis have been proposed [8][14]-[16]. The RTL based SFI has been used either for testing purpose [15] – to derive the fault coverage in early design phases of specific test vectors - , either for reliability evaluation of complex digital systems [7-8][16].

The SFI components for RTL descriptions have been obtained following two approaches. One approach is based on altering the signals within the RTL design [15]**.** This approach does not require the modification of RTL behavioral statements. A second approach is based on the altering the behavioral components, such as the combinational processes [7-8]. These include: replacing the values of conditions in *if* and *case* statements (*stuck-then, stuck-else, dead process, dead clause*), disturbing assignment statements (*assignment control, global stuck-data*), or disturbing operators in expressions (*micro-operation, local stuck-data*), etc. Both approaches can be used to model in an accurate way simple faults, such as stuck-at faults. However, these approaches cannot be used to accurately model transient type of errors or probabilistic faults.

Because SFI performed at a single abstraction level has either the disadvantage of low accuracy or high simulation overhead, several approaches which combine multi-level analysis have been developed [17]-[20].The analysis is performed in a hierarchical manner: low-level analysis for small blocks is performed in order to derive fault models and fault behavior corresponding to higher abstraction layers; the reliability of the entire system is estimated using high level SFI. Works in [17][18][19] propose methodologies to assess the reliability of digital systems described at RTL under Single Event Transient (SET) fault models. For low level analysis, they use analytical methods, such as static timing analysis [18], or SPICE based simulation at transistor level and logic de-rating at gate level [17]. Both methodologies use RTL SFI for the assessment of the entire digital system. A different approach is used in [19]; it is based on SET fault injection for gate level characterization; the critical input combination and its probability is derived for combinational blocks; probabilistic model checking using PRISM is used for deriving the reliability at RTL. Thus, this approach uses SFI for low level characterization, while analytical methods are used at RTL.

III. Probabilistic Data Dependent Faults and Their Gate Level SFI

Probabilistic faults have been indicated as the main reliability issue in sub-powered CMOS devices. An important source of the probabilistic behavior is represented by the logic gate inability to switch in given amount of time. This is due to the wide timing characteristics of sub-powered logic gates, due to the process, voltage and temperature variations. Furthermore, the probabilistic behavior presents a strong data dependency, as the error probability depends on the input switch combination. In [11], four data dependent gate level fault models have been proposed - Fig. 1:

1. *Gate Output Probabilistic (GOP) model* – the logic value of the output of the gate is bit-flipped with a given probability at any moment; the main causes of this phenomenon are the inability of the gate to switch in the given time window or a single event upset affecting the gate
2. *Gate Output Switching (GOS) probabilistic model* – the switching process performed by the logic gate is affected by a probability function, dependent on 3 parameters: supply voltage, temperature and delay.
3. *Gate Output Switching Type (GOST) probabilistic* model – different probabilities are considered for the charging and discharging processes, as a result of different drive strengths of nMOS and pMOS stages
4. *Gate Input Switching Probabilistic (GISP) model* – different probabilities are considered for each possible input transition.

The most accurate fault model is the GISP, as it captures in an accurate way the corresponding input switch combination and its associated error probability. The work in [11] has shown that the proposed methodology can be applied for small and medium logic circuits. However, performing GL analysis using probabilistic fault models is unfeasible for complex circuits. On one hand, GL netlists for complex circuits have millions or more components, thus posing serious computational problems in their simulations. On the other hand, simulating probabilistic faults requires a

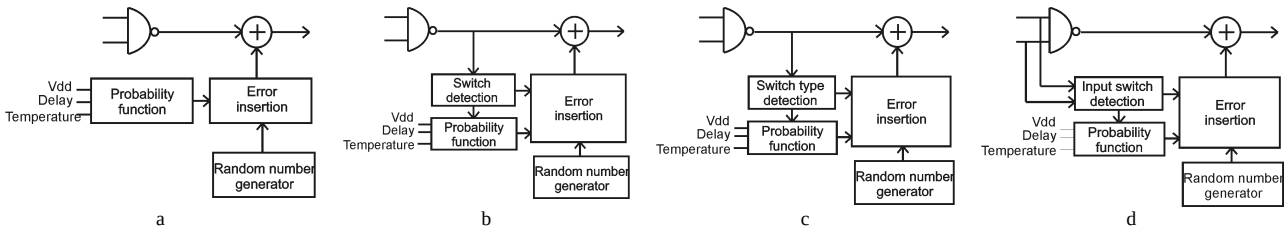

Figure 1. Mutant architecture of NAND gates with the four fault models, according to [11]
(a – GOP, b – GOS, c – GOSP, d – GISP)

high number of simulations in order to be relevant.

Therefore, the probabilistic analysis has to be performed at higher abstraction layers, such as RTL. Another advantage regarding the RTL simulation is represented by the fact that most common synthesizable circuit descriptions are RTL.

IV. Data-Dependent Multi-Level SFI

We propose a hybrid SFI-based reliability evaluation methodology, which aims to combine the advantages offered by GL analysis, namely high accuracy, with the advantages offered by RTL analysis, namely low computational requirements. The GL analysis is based on mutant components, which were developed according to the 4 probabilistic fault models. It aims at capturing the data dependency at block level and to derive the probabilities for the considered blocks. The RTL analysis is based on probabilistic saboteurs, which use the probabilities derived during GL analysis as inputs. The methodology comprises two main phases: SFI performed for GL netlists of each block of the design, while the reliability of the circuits is derived using SFI performed at RTL.

The proposed methodology is depicted in Fig. 2 and was developed according to the following steps:

- Hierarchical block decomposition
- RTL correct simulation
- Logic synthesis
- GL Data dependent SFI
- Probabilistic RTL SFI

Hierarchical block decomposition step is performed in order to partition the system in blocks of lower complexity, which are either full combinational (containing only logic gates), either sequential (containing only storage elements). The goal is to obtain low complexity GL blocks, which can be analyzed using GL SFI.

RTL correct simulation step is performed for the entire system, by applying a given input stimuli set. The goal of this operation is to extract the correct inputs and outputs associated with each block resulted from the decomposition step.

Logic synthesis is performed for each block resulted from the decomposition step; it generates a GL netlist associated with each block. The reliability analysis for these netlists is further performed using GL SFI.

During the *GL data dependent SFI step*, each component of the netlist (logic gate or storage element) is mutated according to the four fault models previously defined: GOP, GOS, GOST or GISP. During the second step, we have extracted the inputs corresponding to each block. The probability of failure for each output signal is obtained by comparing the faulty trace obtained during this step with the gold trace obtained during step 2.

The last step consists in the insertion of *probabilistic saboteurs* on the signals providing the outputs of the RTL blocks. The probabilities used for each saboteur have been derived during the previous step. These saboteurs are used in order to perform RTL SFI for the entire system. The overall probability of failure is computed by comparing the results of the saboteur-based SFI with the golden trace obtained in step 2.

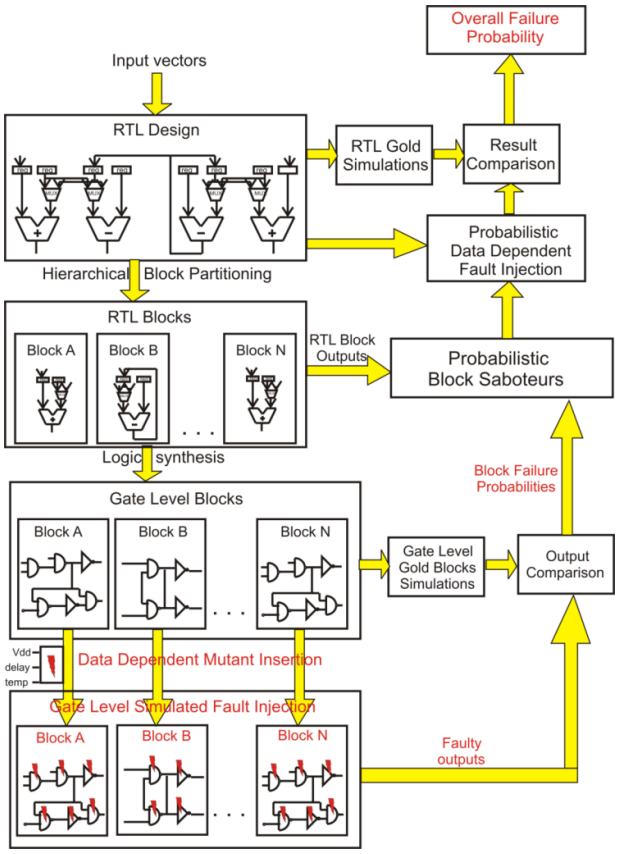

Figure 2. Proposed hybrid GL-RTL SFI

V. Case Studies

We have applied the proposed methodology on a medium size digital circuit (parallel comparator) and on a complex one (AES crypto core).

The reliability evaluation performed on a medium size circuit has been performed in order to establish an accuracy comparison between the hybrid reliability assessment and the GL SFI. The chosen circuit is represented by a parallel comparator, which is the most important component of the CNU processing units within the LDPC decoders. The comparator, depicted in Fig. 3, is presented in [21]-[22] and has the following blocks:

- *Sort module* – this module arranges two pairs of inputs in an ascending mode
- *Compare-select* – this modules has four inputs and outputs the first two minimums

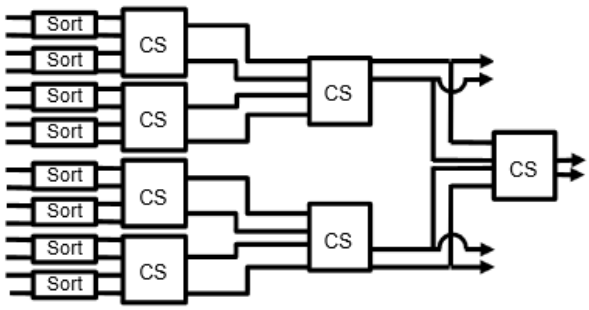

Figure 3. Parallel comparator used in LDPC decoders CNU (CS – compare and select module) [21]-[22]

TABLE I. SIMULATION RESULTS FOR COMPARATOR

Module	Probability of failure	Simulation time	Simulation type
Sort	0.0000%	0.10 ms / run	RTL gold simulation
Sort	0.0000%	0.11 ms / run	Gate level gold simulation
Sort	1.7116%	0.19 ms / run	Gate level FI simulation
Compare Select	0.0000%	0.18 ms / run	RTL gold simulation
Compare Select	0.0000%	0.27 ms / run	Gate level gold simulation
Compare Select	4.8260%	0.62 ms / run	Gate level FI simulation
Comparator	0.0000%	11.7 ms / run	Gate level gold simulation
Comparator	9.3333%	681 ms / run	Gate level FI simulation
Comparator	9.6667%	0.65 ms / run	Gate level + RTL FI simulation

Regarding the probabilities of each logic gate or sequential component, the average error probability of a NAND gate has been considered to be 0.3314%, while the average failure probability for a D flip-flop has been considered to be 0.1251%. This corresponds to the SPICE based simulation results obtained for 45 nm CMOS technology, with supply voltage 0.3 V and a delay constraint of 3 ns per gate and 2.5 ns per flip-flop [9].

Table I presents the results obtained for both hybrid methodology approach and for the GL SFI. Regarding the comparison in terms of accuracy, we observe that the proposed approach has similar results with the GL SFI. Furthermore, the simulation time for our approach is about three orders of magnitude less with respect to the GL SFI.

The second analyzed circuit is represented by an 128-bit Advanced Encryption Standard (AES) crypto-core [23]. We have chosen to apply our fault injection methodology on the AES crypto-core because reliability altering of crypto-cores represents one of the most important side-channel attacks. Therefore, reliability analysis for crypto-cores is highly important.

Fig. 4 depicts the architecture of the AES crypto-core available on OpenCores and used as circuit under test for our experiments [23]. Regarding the behavior of the AES crypto-core, the plain text is represented as 128-bit state, which is altered by repeatedly applying the round transformation 10 times [23]. The final state obtained after the round transformations represent the output cipher text. The 128-bit state is organized as a 4 x 4 matrix of bytes and the round transformation scramble the bytes of the state, either individually, row-wise or column-wise by applying 4 of the 5 above mentioned functions, sequentially: *SubBytes*, *ShiftRows*, *MixColumns* and *AddRoundKey*. The first round is preceded by an initial *AddRoundKey* operation, while the final round is different from the others, because the *MixColumn* operation is omitted. Each function performs transformations on all bytes of the state and there are linear, as well as non-linear reversible operations, allowing the decryption process with their inverses. The only non-linear function of the AES algorithm is *SubBytes*, which substitutes all bytes of the state using table look-up operations and it is usually called S-box. The *ShiftRows* function rotates the rows of the state by an offset, which equals the row index. The *MixColumns* function accesses the state column-wise and interprets a column as a polynomial over GF(28). The *AddRoundKey* function adds a round key to the state and a new round key is obtained in every iteration from the previous round key.

In order to have a measure of the complexity of this core, we provide the synthesis estimates obtained for Xilinx Spartan-6 FPGA devices: 5792 out of 54576 slice registers (10% of the total capacity), 10992 out of 27288 slice LUTs (40% of the total capacity), 29 out of 166 block RAM (25% of the total capacity).

We have applied each step of the fault injection methodology, on the above AES crypto-core. During the first step of the methodology, we have partitioned the circuit under test in 9 functional blocks and we have divided each block in combinational and sequential sub-blocks. A short description of the 9 blocks is presented below:

1. *Block A* represents the AES crypto-chip top module, which has two inputs: the 128-bit key and the 128-bit state. This block firstly performs an exclusive-or on the two vectors and, then, applies the round transformations on the state, using multiple instances of blocks B, C and D.
2. *Block B*, also referred as "expand key" in Fig. 4, performs the expansion operation on the 128-bit key and uses only one instances of block E.
3. *Block C*, also referred as "one round" in Fig. 4, performs XOR operations on the key bytes and instantiates block G.
4. *Block D*, also known as "final round", employs multiple instances of block E.
5. *Block E* is referred as "S4" in Fig. 4 and it substitutes four bytes in a word by calling 4 times the block F module.
6. *Block F* is referred as "S" or "S-box" in Fig. 4, performs a table lookup operation.
7. *Block G* is referred as "table lookup" in Fig. 4, uses the results provided by block H instances.
8. *Block H* is referred as "T" transformation in Fig. 4, uses the results provided by blocks F and I instances.
9. *Block I* is referred as "xS" in Fig. 4, is similar to block F and performs table lookup operations.

During the second step of the methodology, the RTL correct simulation of the entire system is performed for a given set of input vectors. We have extracted the inputs and the outputs for each combinational and sequential sub-block of each block and we have stored them in files containing the correct results for each block, for all the considered runs. Also, during this step, the simulation of block *i*, which contains at least one instance of block *i+1*, generates two files associated to block *i+1*.

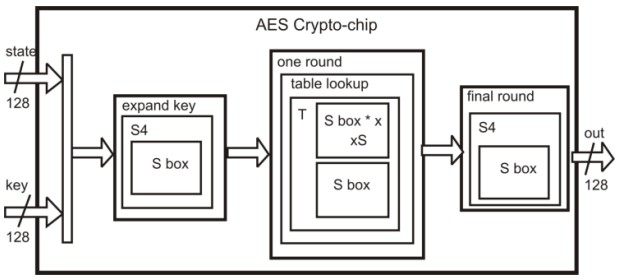

Figure 4. AES crypto-core

TABLE II. AES BLOCK-SPECIFIC FAILURE PROBABILITIES

Module	No. Of input vectors	Output width	Components	Probability of failure	Simulation time	Simulation type
Block I - xS	85435	8	-	9.1250%	33 ms / run	gate level
Block F - S box	3952	8	-	9.0429%	27 ms / run	gate level
Block H - T	85436	32	1 * block F 1 * block I	11.1357%	51 ms / run	gate level + RTL
Block G - table_lookup	3560	128	4 * block H	11.2219%	105 ms / run	gate level + RTL
Block C - one_round	900	128	4 * block G	33.9910%	140 ms / run	gate level + RTL
Block E - S4	1000	32	4 * block F	9.0721%	88 ms / run	gate level + RTL
Block D - final_round	100	128	4 * block E	10.0221%	4 ms / run	gate level + RTL
Block B - expand_key_ 128	1000	128	1 * block E	12.8349%	5.8 ms / run	gate level + RTL
Block A - AES	100	128	10 * block B 9 * block C 1 * block D	50.0625%	93 ms/ run	gate level + RTL

One contains the input vectors applied to all the instances of block *i+1*;the other one contains all the correct outputs which correspond to those inputs.

The third step of the methodology consists in logic synthesis, which has been performed for each of the nine blocks of the design. We have obtained netlists comprised only of 2-inputs NAND gates corresponding to the combinational part and D flip-flops corresponding to the sequential part. The logic synthesis can be performed using commercial tools, such as Cadence Encounter RTL or Synopsis Design Compiler, which generated mapped netlists of the modules, represented in terms of inverters, NAND gates, NOR gates and registers. In order to obtain the desired netlists, each inverter and NOR gate has been further implemented using only NAND gates.

The fourth step consisted in mutant insertion for each NAND gate, according to the GISP model. This model is the most accurate one because it uses 4 probability values, one for each input transition which determines an output transition. SFI has been performed on these mutant-based netlists in order to determine the probability of failure of the blocks situated at the bottom of the design, by confronting the faulty trace with the correct one.

The last step targeted two main objectives: the development of RTL saboteurs and the actual RTL SFI, which has been applied for each level of the module hierarchy. This phase used a bottom-up approach in deriving the probabilistic saboteurs for each component. The probabilities obtained at one level in the hierarchy have been used in order to construct the saboteurs for the next level in the hierarchy.

The results are presented in Table II. The simulations have been performed using the commercial Modelsim simulator, (no optimization option), on a desktop computer with Intel Core i5 processor and 4 GB of RAM. The GL SFI could not be performed on this system. Regarding the simulation time, the entire simulation campaign (consisting of the simulation at gate level and RTL) has taken 131 minutes. This represents a reasonable simulation time for the design containing more than 1 million gates.

Column 3 in Table II depicts the failure probability of each block in the design, as well as the overall failure probability. Due to the hierarchical structure of the design, the 100 input vectors of the crypto-chip can generate thousands or tens of thousands of input vectors for low level blocks. The exact number of input vectors for each considered block in the design is shown in Table II, column 2. We have monitored the simulation time per run. One run represents the simulation of a block for one set of input vectors. Therefore, the number of input vectors in column 2 of Table II is equal to the number of runs performed for a certain block. The total simulation time for a block is equal with the number of inputs multiplied to the simulation time of each run. Although the probability of failure of a single logic gate or flip-flop is extremely low, the resulting probability of failure of one block is quite high, due to the prevalence of XOR operations and bytes scrambling required by the AES algorithm, which facilitates the propagation of faults. Analyzing the results in Table II, we can conclude that the probability of failure of a block increases as we move from the bottom to the top of the design, a fact justified by the increased complexity of upper blocks, which use multiple instances of the lower blocks.

VI. CONCLUSIONS

This paper presents a multi-level GL-RTL SFI methodology for data dependent reliability estimation of digital systems described at RTL. The proposed methodology uses a hierarchical approach: it uses GL data dependent mutant based SFI in order to characterize the RTL components; the reliability analysis of the entire system is obtained using SFI performed at RTL. The proposed approach captures the data dependency using GL simulations for small complexity blocks; these are then used in the RTL data dependent reliability analysis. The proposed methodology consisted on five steps: hierarchical block decomposition, RTL correct simulation, logic synthesis, gate level SFI and RTL SFI. In order to determine the accuracy of the proposed approach with respect to the GL SFI, we have applied the proposed approach on a parallel comparator. The results from the proposed approach have shown good correlation with the GL SFI (9.67% for the proposed vs 9.33% for GL SFI). We have applied the proposed methodology on a complex system, an open-source implementation of a 128 bit AES crypto-core consisting of more than 1 million logic gates. The GL simulation was not possible on the computing platform used.

Using the proposed methodology, the simulations have taken 131 minutes.

The two simulation campaigns have shown both the accuracy, with respect to GL simulations, as well as the scalability for the proposed approach in reliability evaluation of RTL digital circuit descriptions.

REFERENCES

[1] P. Korkmaz, B.E.S Akgul, K. Palem "Energy, Performance, and Probability Tradeoffs for Energy-Efficient Probabilistic CMOS Circuits" *IEEE Trans. On Circuits and Systems I*, vol. 55, Issue 8, 2008, [Online] Available: http://dx.doi.org/ 10.1109/TCSI.2008.920139

[2] B.E.S. Akgul, L.N. Chakrapani, P. Korkmaz, K.V. Palem, "Probabilistic CMOS Technology: A Survey and Future Directions" *Proc. 2006 IFIP Int. Conf. on Very Large Scale Integration*, 2006, pp. 1-6, [Online] Available: [Online]. Available: http://dx.doi.org /10.1109/VLSISOC.2006.313282

[3] K. V. Palem, "Energy aware computing through probabilistic switching: A study of limits", *IEEE Trans. on Computers*, Vol. 54, Issue 9, 2005, pp. 1123-1137, [Online] Available: http://dx.doi.org/ 10.1109/TC.2005.145

[4] V. De "Near-Threshold Voltage design in nanoscale CMOS", *Proc 2013 Design Automation & Test in Europe (DATE)*, 2013, pp. 612-615, [Online] Available: http://dx.doi.org/ 10.7873/DATE.2013.134

[5] H. Khaul, M. Anders, S. Hsu, A. Agarwal, R. Krishnamurthy, S. Bokhar "Near Threshold Voltage Design: Opportunities and Challenges" *Proc. Design Automation Conference (DAC)*, 2012, pp. 1153-1158, [Online] Available: http://dx.doi.org/ 10.1145/2228360.2228572

[6] E. Jenn, J. Arlat, M. Rimen, J. Ohlsson, J. Karlsson "Fault Injection into VHDL Models: The MEFISTO Tool", *Proc. 24th Annual International Symposium on Fault Tolerant Computing (FTCS-24)*, 1994, pp 66-75, [Online] Available: http://dx.doi.org/ 10.1109/FTCS.1994.315656

[7] J. C. Baraza, J. Gracia, D. Gil, P.J. Gil, "Improvement of Fault Injection Techniques based on VHDL Code Modification", *Proc. 10th IEEE International High-Level Design Validation and Test Workshop*, 2005, pp. 19-26, [Online] Available: http://dx.doi.org/ 10.1109/HLDVT.2005.1568808

[8] J. C. Baraza, J. Gracia, S. Blanc, D. Gil, P.J. Gil "Enhancement of Fault Injection Techniques Based on the Modification of VHDL Code" *IEEE Trans. On Very Large Scale Integration (VLSI) Systems*, Vol. 16, Issue 6, 2008, pp. 683-706, [Online] Available: http://dx.doi.org/10.1109/TVLSI.2008.2000254

[9] J Chen, C Spagnol, S Grandhi, E Popovici, S Cotofana, A Amaricai "Linear Compositional Delay Model for the Timing Analysis of Sub-Powered Combinational Circuits" *Proc. 2014 IEEE Annual Symp. On VLSI (ISVLSI)*, 2014, pp. 380-385, [Online] Available: http://dx.doi.org/ 10.1109/ISVLSI.2014.41

[10] M. Merrett, P. Asenov, Y. Wang, M. Zwolinski, D. Reid, et al. "Modelling circuit performance variations due to statistical variability: Monte Carlo static timing analysis" *Proc. 2011 Design Automation and Test in Europe (DATE)*, 2011, pp. 1-4, [Online] Available: http://dx.doi.org/10.1109/DATE.2011.5763329

[11] A. Amaricai, S. Nimara, O. Boncalo, J. Chen, E. Popovici "Probabilistic Gate Level Fault Modeling for Near and Sub-Threshold CMOS Circuits", *Proc. 17th Euromicro Digital System Design*, 2014, pp. 473-479, [Online] Available: http://dx.doi.org/10.1109/DSD.2014.92

[12] S Nimara, A Amaricai, O Boncalo, M Popa "Probabilistic saboteur-based simulated fault injection techniques for low supply voltage interconnects" *Proc. 10th Conf. on PhD Research in Microelectronics (PRIME)*, 2014, pp. 1-4, [Online] Available: http://dx.doi.org/10.1109/PRIME.2014.6872654

[13] S. R. Seward, P. K. Lala, "Fault Injection for Verifying Testability at the VHDL Level", *Proc. International Test Conference (ITC)*, 2003, [Online] Available: http://dx.doi.org/10.1109/TEST.2003.1270833

[14] N. Bombieri, F. Fummi, V. Guarnieri, "Accelerating RTL Fault Simulation through RTL-to-TLM Abstraction", *Proc 16th European Test Symposium (ETS)*, 2011, [Online] Available: http://dx.doi.org/10.1109/ETS.2011.58

[15] P. A. Thaker, "Register-Transfer Level Fault Modeling and Test Evaluation Technique for VLSI Circuits", *Proc. International Test Conference (ITC)*, 2000, [Online] Available: http://dx.doi.org/ 10.1109/TEST.2000.894305

[16] M. Maniatakos, N. Karimi, C. Tirumurti, A. Jas, Y. Makris, "Instruction-Level Impact Comparison of RT- vs. Gate-Level Faults in a Modern Microprocessor Controller", *Proc 27th IEEE VLSI Test Symposium*, 2009, [Online] Available: http://dx.doi.org/ 10.1109/VTS.2009.32

[17] A. Evans, D. Alexandrescu, E. Costenaro, L. Chen "Hierarchical RTL-Based Combinatorial SER Estimation", *Proc. 19th Int. On-Line Testing Symp. (IOLTS)*, 2013, pp. 139-144, [Online] Available: http://dx.doi.org/10.1109/IOLTS.2013.6604065

[18] M. Sonza Reorda, M. Violante "Fault List Compaction through Static Timing Analysis for Efficient Fault Injection Experiments" *Proc. 17th IEEE Symp on Defect and Fault Tolerance in VLSI Systems (DFT)*, 2002, pp. 263-271, [Online] Available: http://dx.doi.org/10.1109/DFTVS.2002.1173523

[19] G. B. Hamad, O. Mohamed, Y. Savaria "Probabilistic model checking of single event transient propagation at RTL level" *Proc. 21st IEEE International Conference on Electronics, Circuits and Systems (ICECS)*, 2014, pp. 471-475, [Online] Available: http://dx.doi.org/ 10.1109/ICECS.2014.7050019

[20] N. Foutris, M. Kaliorakis, S. Tselonis, D. Gizopoulos "Versatile architecture-level fault injection framework for reliability evaluation: A first report" *Proc. 20th Int. On-Line Testing Symp. (IOLTS)*, 2014, [Online] Available: http://dx.doi.org/10.1109/IOLTS.2014.6873686

[21] M. Weiner, B. Nikolic, Z. Zhang " LDPC Decoder Architecture for High-Data Rate Personal-Area Networks", *Proc. Int. Symp on Circuits and Systems (ISCAS)*, 2011, pp. 1784 – 1787, [Online] Available: http://dx.doi.org/10.1109/ISCAS.2011.5937930

[22] Y.S. Park, D. Blaauw, D. Sylvester, Z. Zhang, "*Low-Power High-Throughput LDPC Decoder Using Non-Refresh Embedded DRAM*", IEEE Journal of Solid State Circuits, Vol. 49, Issue 3, 2014, pp. 783-794, [Online] Available: http://dx.doi.org/ 10.1109/JSSC.2014.2300417

[23] 128-bit AES crypto-chip Verilog design, available on Open Cores website: http://www.opencores.org

[24] D. K. Pradhan Fault-Tolerant Computer System Design , Prentice Hall, 1997

[25] C.-C. Lu, S.-Y. Tseng, "Integrated Design of AES (Advanced Encryption Standard) Encrypter and Decrypter" *Proc. IEEE Int. Conf. on Application-Specific Systems, Architectures and Processors (ASAP)*, 2002, pp. 277-285, [Online] Available: http://dx.doi.org/ 10.1109/ASAP.2002.1030726.