

Analyzing an SET at Gate Level using a Conservative Approach

Niels Thole^{*†}, Görschwin Fey^{*†}, Alberto Garcia-Ortiz[‡]

^{*}Institute of Computer Science, University Bremen, Germany
 {nthole, fey}@informatik.uni-bremen.de

[†]Institute of Space Systems, German Aerospace Center, Germany

[‡]Institute of Electrodynamics and Microelectronics, University Bremen, Germany
 agarcia@item.uni-bremen.de

Abstract—Due to the decreasing size of transistors, the probability of transient errors and the variability of the transistor’s characteristics in electrical circuits continues increasing. These issues demand for techniques to check the robustness of circuits and their behavior under transient errors and conservative variability approximations. We present a conservative algorithm to decide if a transient error leads to faulty output of a circuit. Our approach considers logical, timing, and electrical masking as well as variability in the gates. We model these aspects and transient errors at the gate-level ensuring conservativeness of our analysis. In experiments, we compare our algorithm to precise spice-simulations and show the runtime of our implementation on the ISCAS-85 benchmarks.

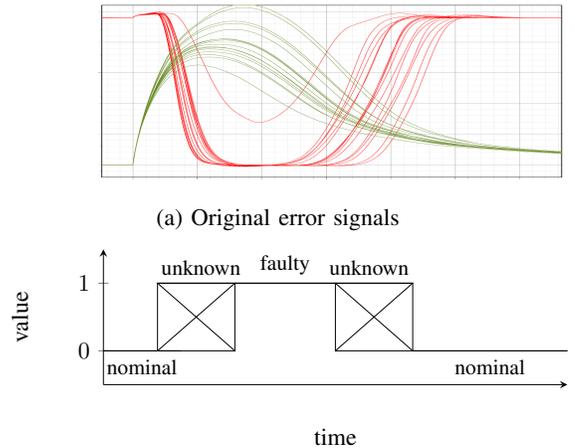
I. INTRODUCTION

New technology decreases size and energy consumption of transistors. While this development enables to create more advanced systems, they become more susceptible to transient faults. External factors like cosmic radiation may induce glitches in the system, which can lead to erroneous behavior. A circuit needs to be analyzed to ensure that no erroneous output is produced under transient faults, i.e., that the circuit is robust. Otherwise vulnerable gates have to be determined.

Our work proposes a model that considers logical, timing, and electrical masking and can model gate variability which is often a result of process variation. The approach determines if a *Single Event Transient* (SET) affects the output of a circuit and leads to erroneous behavior. Our conservative approach aims to ensure that a circuit is robust even under variability of parameters if the approach states that the circuit is robust. Currently, our algorithm is the only tool to symbolically consider variability.

Figure 1a shows different SETs with random variation in the affected gate. The voltage level of a signal temporarily deviates from the nominal value. To analyze the behavior of the circuit, random variation in the timing characteristics of a gate must be taken into account. In our approach we model variations in the gates by using unknown values. Figure 1b shows the resulting abstract model. If the changed signal transmits into the sampling window of an output or a flip flop, an error may occur and our approach returns this information.

This work has been supported by the University of Bremen’s Graduate School SyDe, funded by the German Excellence Initiative.



(b) Model of an SET using a three-valued error signal

Fig. 1: Modeling an SET

Our approach analyses an SET under all possible input combinations using a symbolic analysis based on *Boolean Satisfiability* (SAT).

We validate our model using transistor-level simulations and compare the performance to logic level analysis.

In Section II we summarize related work and identify differences to our work. Section III provides a detailed description of our approach. Section IV shows the results of our experiments. Finally, Section V concludes this work.

II. RELATED WORK

Miskov-Zivanov et al. [5] provide more details for modeling the circuit’s behavior and the effects of an SET. Logical, timing, and electrical masking are considered and the final result is a probability for erroneous output. However, the probabilistic reasoning is computationally expensive.

Omaña et al. [8] provide a mathematical model to analyze the propagation of an SET. While the model is very accurate, it only analyzes electrical masking and does not include logical or timing masking like our algorithm.

WaveSAT [9] is a tool used for automatic test pattern generation. It introduces a *Small-Delay Fault* (SDF) in a certain gate with a certain duration. An SDF is modeled using SAT. The timing behavior of the circuit is described as a SAT

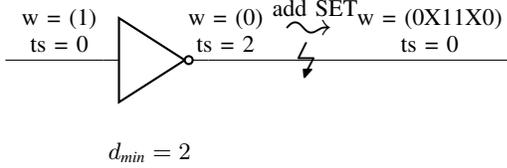


Fig. 2: Generating an SET in a gate

formula and the solution provides input stimuli that detect the given SDF. If the SAT formula is unsatisfiable, the SDF is untestable on the used model. Based on the methods from WaveSAT, Dehbashi et al. [1] developed a tool that finds SDFs under a given erroneous output. The model is similar to WaveSAT and provides a set of gates that could cause the given erroneous behavior due to SDFs. Both, [9] and [1], only take logical and timing masking into account and consider SDFs instead of SETs.

Our work uses a model that is similar to the WaveSAT model. We extend the model to consider electrical masking and variable delays.

The following approaches [2, 3, 4, 10, 11] use abstract models that do not consider timing or electrical masking.

Frehse et al. [2] present an approach to check the effects of a SET. The affected outputs are determined and circuits are classified to be robust if the SET shows no effect, non-robust if the fault becomes visible, or dangerous if the effect is currently not visible but the system is in a different state than it should be. Abstraction for physical characteristics makes the approach efficient.

Han et al. [3] describe a simulation-based analytical approach. They use probabilistic analysis to determine the reliability of a circuit. Due to being simulation-based, the approach does not consider all possible input stimuli.

Leveugle [4] uses classical property checking on mutants of a system to verify that a given error cannot occur.

Seshia et al. [10] detect critical latches by using formal verification. They check if an SET in a latch leads to a behavior that contradicts the specification of the system and finally return a set of critical latches that require additional protection.

Yoeli et al. [11] provide a mathematical description of a circuit at gate level. They analyze the switching between two similar inputs a and b that generate the same output. By using three-valued logic their approach checks if the output can change while switching inputs from a to b .

In summary, some approaches use a coarse abstraction to enable the use on larger circuits while others are very precise but are limited to smaller circuits. Our work aims for an abstraction level in between. We want to check bigger circuits but we do not want to limit the analysis by too many abstractions.

III. CHECKING FOR ROBUSTNESS

We consider the circuit at gate level. This level is more abstract than the transistor level but cannot exactly represent the behavior of a circuit. To ensure the conservative analysis,

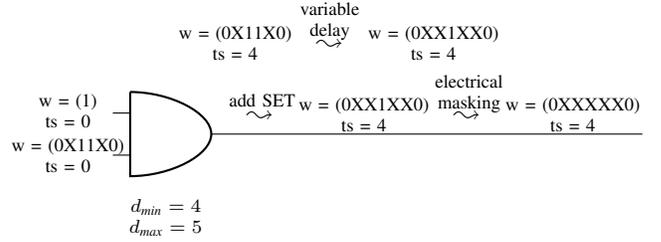


Fig. 3: Propagating a signal considering logical, timing, and electrical masking

we use three-valued logic to describe an approximation of the circuit.

Our approach decides if a given SET can change the output values of the circuit during its sampling window. An assignment of input values that leads to such a change in the output values is called a *counterexample*. If counterexamples exist, our algorithm returns one. Otherwise, the circuit is guaranteed to be robust under the given SET, i.e., the given SET cannot affect the output values for any assignment of input values.

Our algorithm decides the robustness in three steps:

- 1) Define the waveform for every gate iteratively by using the function *propagate*
- 2) Compare the waveform of each output during the sampling window to the nominal value
- 3) Return a counterexample or "circuit is robust" if no counterexamples exist

The function *propagate* defines the waveform of a gate under the given inputs. The execution of *propagate* corresponds to the call of multiple functions: The function *wave* describes the initial waveform of the gate and generated like done in WaveSAT [9]. Since WaveSAT considers constant delays, we use the minimal delay of a gate for that operation. The function *varDelay* describes the variable delay of the gate due to variability and other factors. The SET is given by the function *addSET* and electrical masking is considered in the function *elecMask*. Thus, we define

$$\text{propagate} : V \times (\mathbb{N} \times \text{Var}^*)^* \rightarrow V \times \mathbb{N} \times \text{Var}^*$$

with

$$\text{propagate} = \text{elecMask} \circ \text{addSET} \circ \text{varDelay} \circ \text{wave}.$$

The function *addSET* changes the waveform of the gate that is affected by the SET. In that case, the signals are changed according to the SET, considering the unknown signals at the beginning and end of the SET. An example for this operation is shown in Figure 2. In this example we use explicit values and no symbolic variables like done by our algorithm. In this example, the constant signal with a value 0 is changed to consider the SET. The SET has a length of 4 timesteps and lasts from timestep 1 until 5. During the first and the last timestep of the SET, the signal is set to unknown due to possible variability.

The real delay of a gate depends on different factors, e.g., the inputs or other external influences. To ensure the

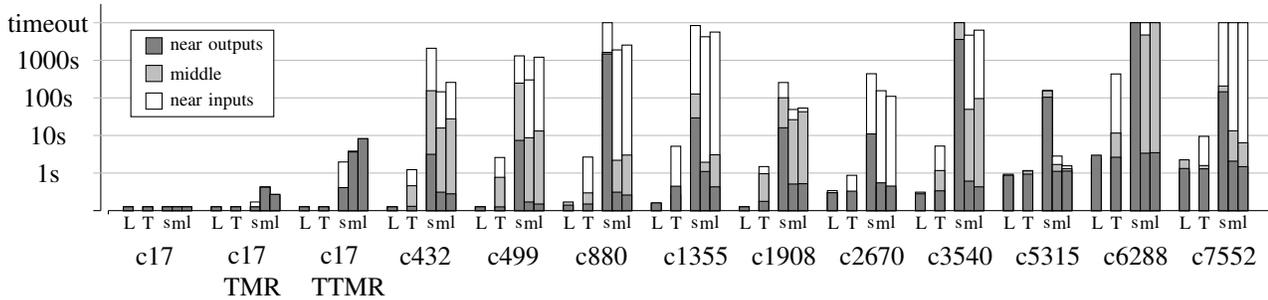


Fig. 4: Runtime of our algorithm

conservatism of our approach, we approximate the delay with a minimal and a maximal delay. If a gate would return different output under different delays between minimal and maximal delay, the output becomes unknown. This is modeled in the function *varDelay*.

Electrical properties of the gates mask short glitches. A glitch is a change of a signal that lasts for a finite time. Afterwards the value of the signal switches back to its original value. The function *elecMask* checks if any glitches are below a threshold that depends on the current gate and removes these glitches if they are short enough.

An example for the whole operation of propagating a signal is shown in Figure 3. First, we consider the variable delay. Since the difference between minimal and maximal delay is $5 - 4 = 1$, we compare each value v_i on the waveform to the following value v_{i+1} . If the values are different, we replace v_i with X. Otherwise, v_i remains unchanged. In a next step, we apply the SET by using *addSET*. Since the gate in this example is not the location of the SET, this function does not change the waveform. Finally, the function *elecMask* applies electrical masking and removes glitches of length 2 or shorter. There is one such glitch in the waveform, it is the 1 in the fourth variable of the waveform. Electrical masking removes this 1 and replaces it with X because the value before and after the glitch is X.

By executing the described steps for each gate, it is possible to represent the whole circuit in form of a SAT formula using three-valued logic. Additionally, we determine the nominal value of each output signal. The nominal value is compared to the output values under the SET during the sampling window by using XOR-operations. All results of these XOR-operations are connected with an OR-operation and the resulting value is given by the variable *overall-error*. In the complete SAT-formula including the determination of nominal values and the described comparison between nominal values and the actual values, the variable *overall-error* is set to 1. Thus, the SAT-formula is only satisfiable if an error is possible under the given SET. In that case a found assignment of variables that satisfies the formula is a counterexample to the robustness of the circuit. If the formula is proven unsatisfiable, the circuit is robust against the given SET.

IV. EXPERIMENTS

We use the ISCAS-85 benchmarks for our experiments. Since those benchmarks do not contain measures for

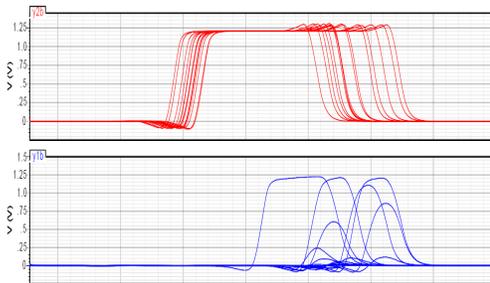


Fig. 5: Output of c17 under a glitch in primary input G3

robustness, in most cases a counterexample is easily found. To further evaluate our algorithm, the circuit c17 has been modified into two robust versions. One version uses *Triple Modular Redundancy* (TMR) to handle SETs. The original circuit is triplicated and a voter decides which output value is returned by using the value of the majority. The other version uses *Timed TMR* (TTMR) similar to [7]. The outputs of the original circuit are delayed by buffers. A voter decides similarly to TMR by using the differently delayed values. This method requires less overhead than TMR but still provides a certain level of robustness.

In a first experiment, we will present the functionality of our algorithm by comparing its results to a spice simulation. Afterwards, we will show the performance of the algorithm by checking its runtime on the circuits from ISCAS-85.

To show the precision of our approach, we compare it to a transistor-level simulation. The simulation is done with the tool Spectre from Cadence and is run on a commercial 65nm technology. In the c17 circuit with added TTMR, we induced an SET into the gate G10 at the front. The spice simulation showed that this transient effect leads to an error if it lasts for 60ps or longer. When our tool assumed fixed delays and was run with timesteps of 5ps, the shortest SET that was classified as a fault had a length of 60ps as well. If we added some variability and added a difference of 5ps between minimal and maximal delay, the shortest SET that was classified had a length of 35ps.

To validate the accuracy and functionality of the approach we test it against Spectre. We use c17 as a test case and perform a detailed Monte-Carlo simulation including on-chip variability for each transistor. The effect of the alpha particles

is modeled as a double exponential current pulse with a parameterizable energy, as done in [6]. When all the inputs are set to 1 and an SET is induced into input G3 the effect is clearly visible in one output as seen in the top graph in Figure 5. However, the effects on the other output vary depending on the variability of the gates as seen on the lower graph. Previous symbolic tools that do not consider variability will not discover the possible error on the second output since the effects of the SET split and overlap without variability. Our algorithm discovers this error.

The following experiments show the runtime of our algorithm. The results of these experiments are shown in Figure 4. We compare our algorithm with electrical masking to our algorithm without electrical masking but with timing masking and a simple logical check. The logical check assigns a single value to each gate and flips the value of the gate affected by the SET. For each circuit, we randomly picked a gate close to the inputs, a gate close to the outputs, and a gate in between as position for the SET. The shade of a bar indicates the position. We inserted a short, a medium, and a long SET ranging from 2 to 10 timesteps. The logical check did not consider the length of the SET, the results are shown by the bars marked with L. For timing masking, the runtimes were similar. We present the average runtime in the bars marked with T. For the algorithm including electrical masking, we present bars for a short SET (s), bars for a medium SET (m), and bars for a long SET (l). We tested every combination of length and position of an SET. When a single experiment took more than 150 minutes, it was aborted.

In the unmodified circuits, a counterexample is always found. The short glitches are exceptions when electrical masking is considered. In case of the short glitches, these are short enough to be masked shortly after triggering and therefore do not change the behavior of the circuit. The TMR version of c17 is not affected by an SET in our tests, which is expected, since TMR can handle SETs inside the circuit. In the TTMR version, we can see that the existence of a counterexample depends on the length of the SET. The buffers can handle the short and middle SET, but not the long SET, since the long SET affects a delayed and a non-delayed signal at the same time.

The checks that include electrical masking take significantly more time. The higher runtime comes from the high number of additional variables required to model the electrical masking. Without electrical masking, our algorithm can handle all circuits in under 509 seconds. Most checks with electrical masking finish within the time limit of 150 minutes. For some of the larger circuits, our algorithm timed out. Additionally, when the effects of the SET split and overlap again, the runtime increases since most optimizations aim for a simplification of signals that do not include this behavior. This increase can be seen in c6288, which is a multiplier, where signals often split and overlap later on. The check for a short SET usually takes more time than the other lengths because the resulting SAT formula is not satisfiable. Finding a counterexample in the other cases is faster than verifying the unsatisfiability.

V. CONCLUSION

We presented an algorithm to conservatively check if a circuit is robust against a given SET. The SET was modeled with an offset, where the value of the signal is unknown due to rising or falling edges. Our input considers the SET under all input stimuli modeling logical, timing, and electrical masking. Different from other existing approaches, we consider the variability of the gates in the circuit, that can lead to different behavior.

To present the runtime and precision of our approach, we compared our implementation to logic level analysis, timing analysis, and transistor-level simulation.

REFERENCES

- [1] Mehdi Dehbashi and Görschwin Fey. SAT-based speed-path debugging using waveforms. In *IEEE European Test Symposium*, pages 1–6, 2014.
- [2] Stefan Frehse, Görschwin Fey, Eli Arbel, Karen Yorav, and Rolf Drechsler. Complete and effective robustness checking by means of interpolation. In *Formal Methods in Computer-Aided Design*, pages 82–90, 2012.
- [3] Jie Han, Hao Chen, Jinghang Liang, Peican Zhu, Zhixi Yang, and Fabrizio Lombardi. A stochastic computational approach for accurate and efficient reliability evaluation. *Computers, IEEE Transactions on*, pages 1336–1350, 2014.
- [4] Regis Leveugle. A new approach for early dependability evaluation based on formal property checking and controlled mutations. In *On-Line Testing Symposium*, pages 260–265, July 2005.
- [5] Natasa Miskov-Zivanov and Diana Marculescu. Multiple transient faults in combinational and sequential circuits: A systematic approach. *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, pages 1614–1627, 2010.
- [6] Kartik Mohanram. Simulation of transients caused by single-event upsets in combinational logic. In *Test Conference, 2005. Proceedings. ITC 2005. IEEE International*, pages 9 pp.–981, 2005.
- [7] Michael Nicolaidis. Time redundancy based soft-error tolerance to rescue nanometer technologies. In *IEEE VLSI Test Symposium*, pages 86–94, 1999.
- [8] Martin Omaña, Giacinto Papasso, Daniele Rossi, and Cecilia Metra. A model for transient fault propagation in combinatorial logic. In *IEEE International On-Line Testing Symposium*, pages 111–115, 2003.
- [9] Matthias Sauer, Alexander Czutro, Ilia Polian, and Bernd Becker. Small-delay-fault ATPG with waveform accuracy. In *Proceedings of the International Conference on Computer-Aided Design*, pages 30–36, 2012.
- [10] Ashwin Seshia, Wenchao Li, and S Mitra. Verification-guided soft error resilience. In *Design, Automation Test in Europe Conference Exhibition*, pages 1–6, 2007.
- [11] Michael Yoeli and Shlomo Rinon. Application of ternary algebra to the study of static hazards. *J. ACM*, pages 84–97, 1964.