Device and Architecture Concurrent Optimization for FPGA Transient Soft Error Rate

Yan Lin and Lei He
Electrical Engineering Department
University of California, Los Angeles
{ylin, lhe}@ee.ucla.edu, http://eda.ee.ucla.edu *

ABSTRACT

Late CMOS scaling reduces device reliability, and existing work has studied the permanent SER (soft error rate) for configuration memory in FPGA extensively. In this paper, we show that continuous CMOS scaling dramatically increases the significance of FPGA chip-level transient soft errors in circuit elements other than configuration memory, and transient SER can no longer be ignored. We then develop an efficient, yet accurate, transient SER evaluation method, called trace based methodology, considering logic, electrical and latch-window maskings. By collecting traces on logic probability and sensitivity and re-using these traces for different device settings, we finally perform device and architecture concurrent optimization considering hundreds of device and architecture combinations. Compared to the commonly used FPGA architecture and device settings, device and architecture concurrent optimization can reduce the transient SER by 2.8X and reduce the product of energy, delay and transient SER by 1.8X.

1. INTRODUCTION

Late CMOS scaling results in the reduction of device reliability [1]. Single-event upset (SEU) due to cosmic rays or high energy particles [2] [3] is one of the most important reliability issues. A transient bit-flip error at a gate output or directly at a latch or flip-flop due to SEU may propagate through the circuit and be captured by a latch or flip-flop, which may affect the circuit functionality for the next several clock cycles and result in chip-level transient soft errors.

Unlike ASICs, SEU may affect configuration SRAMs in FPGAs and may result in *permanent soft error rate*, which cannot be recovered unless re-writing those affected SRAMs [4]. On the other hand, SEU may still result in *transient SER* if the combinational part of an FPGA circuit is affected. Figure 1 compares permanent and transient SER at the sea-level for FPGAs under different ITRS technology nodes [5][1]. SER is measured in number of failures in one billion hours (FIT). It is clear that transient SER is becoming increasingly significant compared to permanent SER and can be no longer ignored. A similar trend has been observed for microprocessors [6]. Moreover, permanent SER detection and correction using various system-level redundancy techniques [4][3] have been extensively studied for FPGAs. In this paper, we therefore consider transient SER optimization for FPGAs and use SER to represent transient SER for the rest of the paper unless otherwise specified.

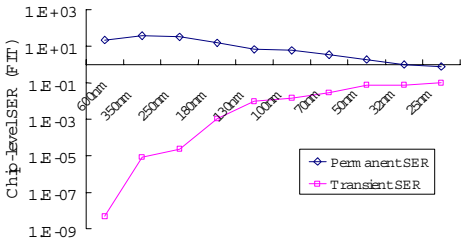


Figure 1: Comparison between permanent and transient sea-level SER for FPGAs (N=8, k=4) across technology nodes using MCNC benchmarks.

SEU strongly depends on device parameters, e.g. supply voltage, V_{dd}, and threshold voltage, V_{th} [2], as well as FPGA architecture parameters, e.g. cluster/LUT sizes. In this paper, we propose the first study on device and architecture concurrent optimization for FPGA SER. Similar to [7], we define *hyper-architecture* as the combination of device (V_{dd} and V_{th}) and architectural parameters (cluster size N and LUT size k). The total number of hyper-architecture combinations can be easily over a few hundred. However, all the existing SER measurement methods [8] and SER simulation algorithms based on fault injection [9] and analytical methods such as [10] are all inefficient to explore the huge hyper-architecture solution space. An accurate, yet extremely efficient, SER evaluation method is required for the concurrent optimization.

We develop a trace based simulation for FPGA chip-level SER considering logic, electrical and latch-window maskings. Compared to the Monte Carlo SER simulation based on fault injection, the trace based simulation is highly accurate and efficient with an average error of 0.04% and a speedup of 1000X for SER analysis. We then perform device and architecture concurrent optimization for FPGA chip-level SER. Overall, device and architecture concurrent tuning leads to a 4.6X difference in SER. Compared to the baseline hyper-architecture similar to a commercial one [11] with delay and energy optimized, the hyper-architecture with minimum SER (min-SER) reduces SER by 2.8X. In general, a larger cluster/LUT size, or a higher V_{dd}/V_{th} may lead to a smaller chip-level SER. We further consider the energy, delay and SER tradeoff and reduce the product of energy, delay and SER (ED·SER) by 1.8X.

The rest of the paper is organized as follows. Section 2 presents the modeling of transient SEU. Section 3 presents the Monte Carlo and trace based SER simulation algorithms. Section 4 discusses the experimental results. We conclude the paper in Section 5.

*This paper is partially supported by NSF grant CCR-0306682 and Actel under UC MICRO program. Address comments to lhe@ee.ucla.edu.

[1] Along with the charge collection slope trend in [6], we use our model and simulation algorithm for transient SER analysis, and the SER model for SRAMs in [2] for permanent SER analysis.

2. TRANSIENT SEU MODELING

2.1 Charge to Voltage Pulse Model

We use the model similar to [6] for transient single event upset (SEU) in combinational circuits. Further study leveraging more accurate models [12] will be conducted. When a particle strikes a sensitive region of a transistor, a current pulse is generated and can be approximated with a one-parameter function as shown in (1),

$$I(t) \propto \frac{Q}{T} \cdot \sqrt{\frac{t}{T}} \cdot e^{-t/T} \quad (1)$$

where Q is the amount of charge and T is the time constant for the charge collection process. This current pulse due to the charge Q then generates a voltage pulse at the transistor output, which can be simulated by SPICE using a transient current source [13].

The shape of the transient voltage pulse can be modeled using rise time, fall time and duration [10], or a Weibull probability density function [14]. In our study, we use *effective duration* [13] to model the voltage pulse shape, where the effective duration is the duration during which the voltage level is greater than $V_{dd}/2$. The shape of a voltage pulse depends on V_{dd}, V_{th}, charge Q and loading capacitance C_{load}. For the simplicity of presentation, we use duration to represent effective duration whenever it is not ambiguous.

As in SRAMs, the critical charge, Q_{crit}, for combinational circuits is the smallest charge that can generate a voltage pulse with duration greater than zero [6]. A larger charge Q may result in a voltage pulse with longer duration. For combinational circuits, we also measure Q_{max}, which can generate the voltage pulse with the longest duration. We use SPICE simulation for SER library pre-calibration. Given a charge Q, the SER induced by any charge no smaller than Q can be calculated as,

$$SER(Q) = F \cdot A \cdot K \cdot e^{-Q/Q_s} \quad (2)$$

where F is the neutron flux, A is the transistor drain area, K is a technology independent constant and is the same for all device settings, and Q_s is the charge collection slope. A transistor with a larger Q_s is more effective in collecting charge and has a larger SER. Q_s depends on V_{th} (due to channel doping density) and V_{dd}. A transistor with higher V_{th} due to a larger channel doping density may enhance the channel resistance, and thus reduce the collection process and result in a lower Q_s [15]. We use the models in [2] with linear interpolation to calculate Q_s for different V_{dd} and V_{th}. Note that PMOS and NMOS transistors have different Q_s. $SER(Q)$ in (2) depends on the current logic value of the gate output. The probability of a voltage pulse with duration W due to the charge between Q and $Q + \Delta Q$ can be calculated as,

$$P_{charge}(W) = SER(Q) - SER(Q + \Delta Q) \quad (3)$$

The probability of a voltage pulse with the maximum duration W_{max} due to any charge larger than Q_{max} can be calculated as,

$$P_{charge}(W_{max}) = SER(Q_{max}) \quad (4)$$

While not presented in the paper, our simulation results show that Q_{crit} and Q_{max} depend on C_{load} and V_{dd}, but not V_{th}. A larger C_{load} or V_{dd} implies greater energy (Q) that is required to flip the gate output. On the other hand, the maximum duration, W_{max}, of the voltage pulse depends on C_{load}, V_{dd} and V_{th}. A larger C_{load}, a larger V_{th} or a smaller V_{dd} may result in a larger W_{max}. In the other words, a gate with a larger delay may have a larger W_{max}.

2.2 Masking Mechanisms

A transient bit-flip error at a gate output may not affect a combinational logic circuit unless this transient voltage pulse can propagate through the circuit and be captured by a memory circuit, e.g. a latch or a flip-flop. There are several masking mechanisms including *logic masking*, *electrical masking* and *latch-window masking* for combinational circuits. Logic masking occurs when a transient voltage pulse at one input of a gate is *blocked* by this gate, i.e. the logic value of the gate output is completely determined by its other inputs under this particular input vector. Logic masking depends on the circuit input vector and also circuit topology.

Electrical masking occurs when a transient voltage pulse is attenuated in both amplitude and duration by the subsequent logic gate due to the electrical properties of the gate. It has been shown in [13] that the effective duration of a voltage pulse can be used to capture the voltage pulse characteristics in a wide range of logic gates and charges. In addition, the duration degradation of a transient voltage pulse is directly affected by its own duration, W_{in}, and the fanout gate delay, T_{dly}. We model the duration of the output voltage pulse, W_{out}, based on W_{in} and T_{dly} as follows,

$$W_{out} = W_{in} \cdot f(\frac{W_{in}}{T_{dly}}) \quad (5)$$

where $f(W_{in}/T_{dly})$ is a function of the ratio between W_{in} and T_{dly}. We use SPICE simulation to measure W_{out} with various W_{in}, T_{dly} and a wide range of logic gates, and then extract an empirical piece wise linear (PWL) function.

After successfully propagating through the combinational circuit considering logic and electrical maskings, a transient voltage pulse can only be captured by a latch or a flip-flop during a small window around its closing clock edge, called a *latch window*. The size of this latch window is the minimum duration of a pulse that can be latched. In our study, we use the latch-window masking model from [6]. The probability that a voltage pulse at the latch input can be captured by this latch is calculated as follows,

$$P_{latch}(W) = \begin{cases} 0 & \text{if } W < L \\ \frac{W-L}{C} & \text{if } L \le W \le C + L \\ 1 & \text{if } W > C + L \end{cases} \quad (6)$$

where W is the duration of the voltage pulse at the latch input, L is the size of latch window, and C is the clock period. When W is smaller than L, the probability of a soft error, i.e. being captured by the latch, is zero. On the other hand, a pulse with duration larger than $C + L$ can cover one full latch window and hence has a probability of one for a soft error. Since pulse arrival times are uniformly distributed in a clock cycle, the probability that a pulse with any intermediate duration may result in a soft error is the linear interpolation between two extreme cases.

3. CHIP-LEVEL SER SIMULATION

3.1 Monte Carlo Simulation

We develop the Monte Carlo SER simulation based on fault injection. An input vector is randomly generated for primary inputs in each clock cycle. Based on this input vector, we traverse the circuit in the topological order and evaluate the output logic value of each gate. The critical charge Q_{crit} and the maximum charge Q_{max} can be obtained in the pre-calibrated SER library given the current device setting, i.e. V_{dd} and V_{th}, and C_{load} of this gate. We then evenly sample m points between Q_{crit} and Q_{max}, where the i_{th} point has the charge Q_i as follows,

$$Q_i = Q_{crit} + i \cdot \frac{Q_{max} - Q_{crit}}{m - 1} \quad 0 \le i \le m - 1 \quad (7)$$

The duration, W_i, of the generated transient voltage pulse due to Q_i can also be obtained from the pre-calibrated library. The probability

for such a charge Q_i being collected is $P_{charge}(W_i)$ that can be calculated by (3) or (4) considering the current logic value of the gate. We then propagate the voltage pulse with duration W_i through the fanout cone of the gate considering logic masking and electrical masking mechanisms. If this voltage pulse successfully propagates through the circuit and reaches a flip-flop input, the probability of this voltage pulse being captured by the flip-flop and resulting a soft error is $P_{latch}(W_i')$, which can be calculated by (6). W_i' is the voltage pulse duration at the flip-flop input. The chip-level SER of a voltage pulse with duration W_i due to charge Q_i for one gate within one clock cycle can be calculated as follows,

$$SER_{gate}(W_i) = P_{charge}(W_i) \cdot \sum_{latch} P_{latch}(W_i') \qquad (8)$$

where all the flip-flops within the fanout cone of the gate are considered. The average chip-level transient SER is contributed by all gates in all clock cycles and can be calculated as follows,

$$SER_{chip} = \frac{\sum_N \sum_{gate} \sum_i SER_{gate}(W_i)}{N} \qquad (9)$$

where N is the total number of simulated input vectors. Figure 2 shows the the Monte Carlo simulation algorithm and the overall complexity is $O(Nm|V|(|V|+|E|))$, where N is the total number of simulated input vectors, m is the number of charge samples, $|V|$ is the number of gates and $|E|$ is the number of edges in the circuit.

```
Monte Carlo Simulation:
For each input vector {
  Traverse the circuit and evaluate logic value for each gate;
  For each gate g the the circuit{
    For each W_i associated with Q_i{
      Calculate P_charge considering logic value of g;
      Traverse the fanout cone of g and accumulate SER_N;
    }
  }
}
SER_chip = SER_N / N;
```

Figure 2: The Monte Carlo simulation based on fault injection for the chip-level transient SER.

3.2 Trace Based Methodology

The above Monte Carlo based simulation has high complexity with expensive runtime and is impractical for device and architecture concurrent optimization. To enable this concurrent SER optimization, we propose the efficient, yet accurate, trace based methodology. We first profile the benchmarks and collect the statistical information, called *trace information*. The trace information is invariant when the device setting is changed. We then vary the circuit level model, e.g. gate delay, critical charge Q_{crit} etc., based on the device setting. Using the device independent trace information and the device dependent circuit level model as the inputs, we develop the trace based SER simulation to estimate the chip-level SER for each hyper-architecture considering the combinations of device and architecture settings. A similar methodology has been applied to micro-processors [16] and FPGAs [7] for power optimization. The details of the trace based SER simulation methodology are discussed as follows.

3.2.1 Trace Collection

We collect two types of trace information. The first one is *logic probability*, i.e. the probability that the stable logic value of each gate output is logic '1'. The second type of trace information is *average logic sensitivity*. The logic sensitivity is calculated for each input of each gate. Given a gate with stable inputs and output, the gate output is sensitive to an input, i.e. this input has a sensitivity of one, if changing the logic value of this input will result in the change of the gate output logic value. For each input vector, we first traverse the circuit, and evaluate the logic value and logic sensitivity for each gate output and input, respectively. We then calculate the logic probability for each gate output and the average logic sensitivity for each gate input after simulating N input vectors. The complexity of this trace collection process is $O(N(|V|+|E|))$, where N is the total number of input vectors. Note that the trace only needs to be collected once under one device setting and can be reused during device and architecture concurrent optimization. Only steady state logic information is considered in the trace collection process and it is easy to see that the trace is independent of timing model and device setting. Therefore we have the following theorem.

THEOREM 1. *The trace information including logic probability and average logic sensitivity for gate outputs and inputs are independent of device setting and technology.*

3.2.2 Chip-level SER Analysis

Given the collected trace information and the circuit level model, we then develop the trace based SER simulation algorithm. Similar to the Monte Carlo simulation, for each gate g in the circuit we obtain the critical charge Q_{crit} and maximum charge Q_{max} from the pre-calibrated library, and evenly sample m charge points between Q_{crit} and Q_{max} using (7). We then obtain the duration W_i of the voltage pulse due to the i^{th} charge Q_i. Instead of directly calculating the probability of such a charge Q_i by (3) or (4) using the logic value of gate g, we calculate the probability of charge Q_i with the logic probability $P_g(1)$ of gate g as,

$$P'_{charge}(W) = P_g(1) \cdot P_{charge}(W)|_{(g=1)} + (1-P_g(1)) \cdot P_{charge}(W)|_{(g=0)} \qquad (10)$$

We then propagate the voltage pulse with duration W_i through the fanout cone of gate g. During propagation, the average logic sensitivity is considered to model logic masking statistically. Suppose a voltage pulse with duration W_{in} and probability P_{in} arrives at the i^{th} input of a gate g'. With a particular input vector, this transient voltage pulse may be logically blocked if the gate output is insensitive to the i^{th} input. In other words, the probability of the output voltage pulse, P_{out}, is zero. Without using input vectors, we degrade the probability P_{in} of the voltage pulse at the gate input by a factor of the average logic sensitivity to calculate P_{out} for the voltage pulse at the gate output as follows,

$$P_{out} = P_{in} \cdot Average\_Sensitivity(g', i) \qquad (11)$$

where $Average\_Sensitivity(g', i)$ is the average logic sensitivity of gate g' to its i^{th} input. The duration of the voltage pulse at gate output, W_{out}, is calculated based on W_{in} and the gate delay T_{dly} considering electrical masking as discussed in Section 2.2. If the voltage pulse duration degrades to zero due to electrical masking, i.e. $W_{out} = 0$, we set P_{out} to zero and stop propagating through the fanout cone of this gate. At the first level of propagation from gate g, the voltage pulse at the output of gate g has a duration of W_i with probability of P'_{charge} as calculated by (10). If the voltage pulse successfully reaches a flip-flop input, we then further consider the latch masking using (6) and accumulate the chip-level SER.

Figure 3 presents the high-level algorithm of the trace based SER simulation. The complexity of the trace based simulation is $O(m|V|(|V|+|E|))$, where m is the number of charge samples. The overall complexity of the trace based methodology including trace collection is $O(N(|V|+|E|)+m|V|(|V|+|E|))$, which is still smaller than the complexity of the Monte Carlo simulation.

Trace Based Algorithm:
For each gate g the the circuit{
  For each W_i associated with Q_i {
    Calculate P'_{charge} considering logic probability of g;
    Traverse the fanout cone of g and accumulate SER_{chip};
  }
}

Figure 3: The trace based SER simulation.

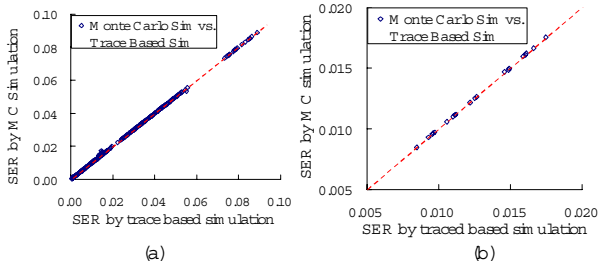
4. EXPERIMENTAL RESULTS

In this section, we conduct the experiments on the largest MCNC benchmarks [17]. We use the Berkeley predictive device model [18] at ITRS [5] $65nm$ technology node. The island style architecture [19] is used in our study. We assume the sea-level SER with the same flux F as in [2]. SER is measured in number of failures in one billion hours (FIT). The baseline hyper-architecture uses the same cluster and LUT sizes as those used by the Xilinx Virtex-II [11] (cluster size of 8, LUT size of 4), V_{dd} suggested by ITRS [5] (0.9v), and V_{th} (0.3v) that is optimized for the above architecture and V_{dd} considering energy and delay product [7]. The optimization ranges are {N=6, 8, 10, 12}, {k=3, 4, 5, 6, 7}, {V_{dd}=0.8v, 0.9v, 1.0v, 1.1v} and {V_{th} = 0.2v, 0.25v, 0.3v, 0.35v, 0.4v}.

4.1 Validation of The Trace Based Simulation

We assume V_{dd} of 0.9v and V_{th} of 0.3v as suggested by ITRS in this section. We use *fpgaEVA-LP2* [20] to map a benchmark to an FPGA chip and extract the delay and parasitics annotated gate-level netlist, which is the input of both the Monte Carlo based simulation and the traced based one. In our study, we perform the Monte Carlo simulation for 1000 input vectors such that the fluctuation of the average SER in the last 50 input vectors is less than 0.5%.

We compare the SER from the trace based simulation with the the Monte Carlo based one in Figure 4. Each of 20 MCNC benchmarks is mapped to each of 20 FPGA architectures. Figure 4 (a) compares the two algorithms for each individual benchmark, i.e. 400 comparisons. The maximum absolute difference between the SER from the two algorithms is 5.8% while the average difference is only 0.04%. Figure 4 (b) compares the two algorithms for each architecture, where the SER of one architecture is calculated as the geometric mean of 20 benchmarks for this architecture. The maximum difference between the SER from the two algorithms for one architecture is only 0.39% while the average difference is 0.03%.



(a) (b)

Figure 4: The comparison of the chip-level transient SER between the Monte Carlo (MC) simulation and the traced based simulation.

In addition, it takes 48 hours of runtime to perform the Monte Carlo based simulation for one architecture with 20 benchmarks. On the other hand, it only takes 5 hours to collect the trace information as defined in Section 3.2 and 3 minutes to perform trace based simulation for the same set of benchmarks. Note that the trace information only needs to be collected once and can be re-used in device and architecture concurrent optimization. The amortized speedup of the trace based SER simulation is 1000X compared to the Monte Carlo based simulation. It is clear that the trace based SER simulation is highly efficient and accurate compared to the Monte Carlo based simulation.

4.2 Impact of Architecture and Device Tuning

We use the trace based simulation to perform device and architecture evaluation for SER optimization. Starting with the baseline hyper-architecture {N=8, k=4, V_{dd}=0.9v, V_{th} =0.3v}, we first study the impact of architecture tuning on SER in Figure 5. From this figure, we have the following observation.

OBSERVATION 1. *In general, a larger cluster size N or a larger LUT size k may lead to a smaller chip-level transient SER. In addition, tuning cluster size N or LUT size k has a similar impact on SER.*

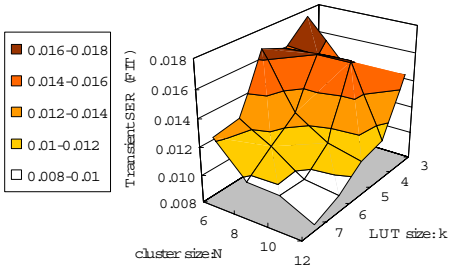


Figure 5: The impact of architecture tuning on chip-level SER with a fixed device setting, i.e. V_{dd} = 0.9v, V_{th} = 0.3v.

The impact of tuning cluster (or LUT) sizes on SER can be obtained by calculating the ratio between the maximum and minimum SER with a fixed LUT (or cluster) size, but sweeping all possible cluster (or LUT) sizes. On average, tuning cluster size in {N=6, 8, 10, 12} leads to a 1.6X difference in SER while tuning LUT size in {k=3, 4, 5, 6, 7} leads to a 1.4X difference in SER. Overall, with this device setting (V_{dd} = 0.9v and V_{th} = 0.3v), architecture tuning leads to a 2.1X difference in SER, i.e. the architecture (N=6, k=3) or (N=6, k=5) has a maximum SER of 0.017 FIT while the architecture (N=12, k=7) has a minimum SER of 0.008 FIT.

Based on the optimized architecture (N=12, k=7), we then study the impact of device tuning on SER in Figure 6. From this figure, we have the following observation.

OBSERVATION 2. *In general, a higher V_{dd} or a higher V_{th} may lead to a smaller chip-level transient SER. In addition, tuning V_{dd} or V_{th} has a similar impact on SER.*

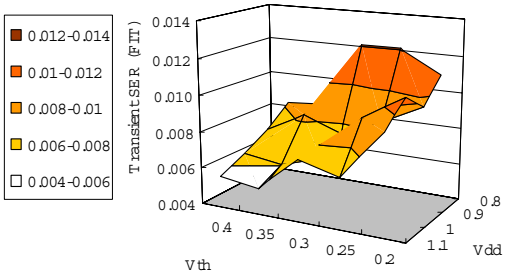


Figure 6: The impact of device tuning on chip-level SER with a fixed FPGA architecture (N=12, k=7).

Similar to the impact of tuning cluster (or LUT) sizes, we study the impact of tuning V_{dd} (or V_{th}) on SER. On average, tuning V_{dd} in {0.8v, 0.9v, 1.0v, 1.1v} leads to a 1.6X difference in SER while tuning V_{th} in {0.2v, 0.25v, 0.3v, 0.35v, 0.4v} also leads to a 1.6X difference in SER. Tuning V_{dd} or V_{th} has a similar impact on SER. Overall, with the architecture (N=12, k=7), device tuning leads to an 2.2X difference in SER, i.e. the device setting (V_{dd}=0.8v, V_{th}=0.25v) has a maximum SER of 0.012 FIT while the device setting (V_{dd}=1.1v, V_{th}=0.35v) has a minimum SER of 0.0054 FIT. Moreover, compared to architecture tuning, device tuning has a similar impact on SER.

4.3 Device and Architecture Concurrent Optimization

There are three methods to perform device and architecture optimization. In the first two methods, we can first optimize architecture (or device) then optimize device (or architecture) given the optimized architecture (or device). In the third method, we optimize architecture and device concurrently, called *concurrent* method. While not presented here, our experimental results show that only the *concurrent* method can guarantee the global optimal solution. Due to the efficient trace based simulation algorithm, the overall runtime including the trace collection process of the *concurrent* method is affordable (25 hours). The concurrent optimization leads to a 4.6X difference in SER, i.e. hyper-architectures {N=12, k=6, V_{dd}=1.1v, V_{th}=0.4v} and {N=6, k=3, V_{dd}=0.8v, V_{th}=0.2v} obtain a minimum SER of 0.0052 FIT and maximum SER of 0.024 FIT within the whole solution space, respectively.

	hyper-architecture {N, k, V_{dd}, V_{th}}	ED (nJ·ns)	SER (FIT)	ED·SER (nJ·ns·FIT)
baseline	{8, 4, 0.9v, 0.3v}	1.2X	2.8X	1.8X
Min-ED	{12, 4, 0.9v, 0.25v}	14.7	2.2X	1.2X
Min-SER	{12, 6, 1.1v, 0.4v}	3.3X	0.0052	1.8X
Min-ED·SER	{10, 6, 1.0v, 0.3v}	1.2X	1.4X	0.14

Table 1: Comparison between the baseline, min-ED, min-SER and min-ED·SER hyper-architectures.

We further consider the energy, delay and SER tradeoff during the concurrent optimization. We use *Ptrace* [7] to estimate energy and delay for each hyper-architecture. Table 1 compares the baseline, min-ED, min-SER and min-ED·SER hyper-architectures in detail. In this table, ED product, SER and ED·SER product in each column are normalized to the minimum corresponding value, respectively. The min-SER hyper-architecture reduces SER by 2.8X compared to the baseline but obtains a 3.3X larger ED product compared to the min-ED one. As a result, the min-SER and baseline hyper-architectures obtain the same ED·SER product. On the other hand, the min-ED·SER hyper-architecture obtains a 1.4X larger SER and 1.2X larger ED compared to the min-SER and min-ED hyper-architectures, respectively. Compared to the baseline hyper-architecture, the min-ED·SER one obtains the same ED product but reduces SER by 2X (2.8X/1.4X=2X). Moreover, the min-ED·SER hyper-architecture obtains a 1.8X, 1.2X or 1.8X smaller ED·SER product compared to the baseline, min-ED and min-SER hyper-architectures, respectively. The min-ED·SER hyper-architecture achieves the best energy, delay and SER tradeoff.

5. CONCLUSIONS AND DISCUSSIONS

In this paper, we have shown that continuous CMOS scaling dramatically increases the significance of FPGA chip-level transient soft errors in circuit elements other than configuration memory, and transient SER can no longer be ignored. We have developed an efficient, yet accurate, trace based simulation for FPGA chip-level transient SER considering logic, electrical and latch-window maskings. The statistical trace information on logic probability and logic sensitivity is collected once for a given set of benchmark circuits and can be reused during optimization.

We have also performed device and architecture concurrent optimization for FPGA chip-level SER. Overall, device and architecture concurrent tuning leads to a 4.6X difference in SER. Compared to the baseline hyper-architecture similar to a commercial one [11] with delay and energy optimized, the hyper-architecture with minimum SER (min-SER) reduces SER by 2.8X. In general, a larger cluster/LUT size or a higher V_{dd}/V_{th} may lead to a smaller chip-level SER. We have further considered the energy, delay and SER tradeoff during the concurrent optimization and reduce the product of energy, delay and SER (ED·SER) by 1.8X.

In the future, we will explore more device parameters and study early stage technology optimization considering the impact of FPGA synthesis and architecture leveraging the trace based methodology.

6. REFERENCES

[1] S. Borkar, "Electronics Beyong Nano-scale CMOS," in *Proc. Design Automation Conf.*, July 2006.

[2] P. Hazucha and C. Svensson, "Impact of cmos technology scaling on the atmospheric neutron soft error rate," *IEEE Trans. on Nuclear Science*, 2000.

[3] W. Heidergott, "SEU tolerant device, circuit and process design," in *Proc. Design Automation Conf.*, June 2005.

[4] G. Asadi and M. Tahoori, "Soft error rate estimation and mitigation for SRAM-based FPGAs," in *ISFPGA*, Feb 2005.

[5] International Technology Roadmap for Semiconductor in *http://public.itrs.net/*, 2003.

[6] P. Shivakumar and et al, "Modeling the impact of device and pipeline scaling on the soft error rate of process elements," in *CS Dept, the Univ. of Texas at Austin, Technical Report 2002-19.*

[7] L. Cheng, P. Wong, F. Li, Y. Lin, and L. He, "Device and architecture co-optimization for FPGA power reduction," in *DAC*, June 2005.

[8] A. Lesea and et al, "The rosetta experiment: atmospheric soft error rate testing in differing technology FPGAs," *IEEE Transactions on Device and Materials Reliability*, 2005.

[9] A. Frantz and et al, "Evaluation of SEU and crosstalk effects in network-on-chip switches," in *Proceedings of the 19th annual symposium on Integrated circuits and systems design*, 2006.

[10] B. Zhang and M. Orshansky, "Symbolic simulation of the propagation and filtering of transient faulty pulses," in *Workshop on system effects of logic efforts*, April 2005.

[11] Xilinx Corporation, "Virtex-II 1.5v platform FPGA complete data sheet," July 2002.

[12] K. Warren and et al, "Predicting Thermal Neutron-Induced Soft Errors in Static Memories Using TCAD and Physics-Based Monte Carlo Simulation Tools," *IEEE Electron Device Letters*, vol. 28, pp. 180–182, Feb 2007.

[13] G. Wirth and et al, "Single event transients in combinational circuits," in *Intl. Symp. on Integrated circuits and system design*, September 2005.

[14] A. Kasnavi and et al, "Analytical modeling of crosstalk noise waveforms using weibull function," in *ICCAD*, Nov 2004.

[15] P. Roche and G. Gasiot, "Impacts of front-end and middle-end process modifications on terrestrial soft error rate," *IEEE Transactions on Device and Materials Reliability*, Sept 2005.

[16] J. Eble and et al, "A generic system simulator (GENESYS) for ASIC technology andarchitecture beyond 2001," in *The International ASIC Conference and Exhibit*, Sept. 1996.

[17] S. Yang, "Logic synthesis and optimization benchmarks, version 3.0," tech. rep., Microelectronics Center of North Carolina (MCNC), 1991.

[18] U. of Berkeley Device Group, "Predictive technology model," in *http://www.device.eecs.berkeley.edu/ ptm/mosfet.html*, 2002.

[19] V. Betz, J. Rose, and A. Marquardt, *Architecture and CAD for Deep-Submicron FPGAs.* Kluwer Academic Publishers, 1999.

[20] Y. Lin, F. Li, and L. He, "Power modeling and architecture evaluation for FPGA with novel circuits for Vdd programmability," in *ISFPGA*, Feb 2005.