Output Remapping Technique for Soft-Error Rate Reduction in Critical Paths

Qian Ding, Yu Wang, Hui Wang, Rong Luo, Huazhong Yang
Dept. of EE, Tsinghua Univ., Beijing, 100084, P.R. China
dingq03@mails.tsinghua.edu.cn, { yu-wang, wangh, luorong, yanghz}@tsinghua.edu.cn

Abstract

It is expected that the soft error rate (SER) of combinational logic will increase significantly. Previous solutions to mitigate soft errors in combinational logic suffer from delay penalty or area/power overhead. In this paper, we proposed an output remapping technique to reduce SER of critical paths. Experimental results show up to about 20X increase in $Q_{critical}$. So the SER is reduced significantly. This method does not introduce any delay penalty. The area/power overhead is limited as well. The output remapping method is based on our novel glitch width model. The analysis shows that output remapping technique works well along with technology scaling.

1. Introduction

Alon[1]g with the technology scaling, the smaller node capacitance and lower supply voltage make the soft error a big concern. Alpha particles, fast neutrons, and thermal neutrons are three major ground level soft error donors [1, 2, 3]. For memory elements, if particle strike deposited charge is more than a minimum value ($Q_{critical}$), the stored data will be damaged and a soft error occurs. Combinational elements were considered much less vulnerable to soft errors. However, because of technology scaling *SER* of combinational circuits is expected to increase significantly [2].

A common method to mitigate soft error is Triple modular redundancy (TMR), which results in 200% overhead of area and power [4]. Time redundancy and partial duplication induces less overhead, but still add additional delay [5, 6]. Gate sizing has been proposed to reduce *SER* [7], but increased gate size results in area and power overhead. Optimal assignment of supply voltage, threshold voltage, gate size and additional load capacitance enhance the electrical masking effect and reduce *SER* [8]. This technique induces area and power overhead as well. Latching-window masking through flip-flop selection [9] can also be used to reduce *SER*.

In this paper, we focus on the combinational outputs and propose an output remapping technique to reduce *SER* for the following reasons. First, glitches need to be propagated to the combinational output to cause a soft error. Second, if the pulse width is smaller than the logic cell propagation delay t_p, the glitch can not be propagated. Third, the narrow glitches determine *SER*. Therefore we can replace some gates connected to the outputs with other complex logic gates that have longer delay, and then narrow glitches will be filtered out at the outputs. We name this technique output remapping. There are two questions about this method.

1. Is the complex gate fast enough to replace the simple gates? We will show in Section 4 that sometimes static logic is fast enough. And for those gates which are not fast enough, we could choose dynamic logic.
2. Is this method still applicable when the technology scales down and the propagation delay of the complex gate becomes smaller? We will answer this question in Section 4.

The paper is organized as follows. Section 2 introduces our soft error analysis model and our glitch width model. Section 3 presents our output remapping technique to reduce *SER*. Experimental results are presented and discussed in Section 4. We summarize the paper in Section 5.

2. Soft Error Analysis Model

Soft error models including glitch generation model and glitch propagation model will be introduced in this section. We propose a novel glitch width model which shows why we can expect the glitch width scales down with technology scaling.

2.1. Transient Pulse Generation Model

[1] This project is supported by NSFC 90707002, 60506010. It is also partly supported by Basic Research Foundation of Tsinghua SIST (No SIST2017).

0-7695-3117-2/08 $25.00 © 2008 IEEE
DOI 10.1109/ISQED.2008.86

Particles that strike the silicon bulk will inject a very short current pulse at the circuit node. Equation (1) can be used to estimate this effect [10].

$$I(t) = I_{peak} \times (e^{-t/\tau_\alpha} - e^{-t/\tau_\beta}) \qquad (1)$$

I_{peak} is the amplitude of the pulse, τ_α and τ_β are technology dependent time constants. The strength of the current injection induced voltage pulse is described by the pulse width (t_w) measured at 0.5 V_{dd}. If the amplitude of the current pulse (I_{peak}) is more than a minimum value, t_w will be larger than 0. t_w will increase with I_{peak}. Soft error rate can be estimated using an empirical model [1], which describes the relationship between SER and minimum charge deposited by the particle strike ($Q_{collected}$):

$$SER \propto N_{flux} \times CS \times e^{\frac{-Q_{collected}}{Q_S}} \qquad (2)$$

Where N_{flux} is the neutron flux, CS is the drain area struck by neutron flux, Q_s is the charge collection efficiency. $Q_{collected}$ is derived from (1):

$$\begin{aligned} Q_{collected,i} &= \int I(t) \\ &= I_{peak} \times (\tau_\alpha - \tau_\beta) \end{aligned} \qquad (3)$$

The $Q_{collected}$-t_w characteristic of inverters is illustrated in figure 1. Please notice that t_w increase significantly after an inflexion point.

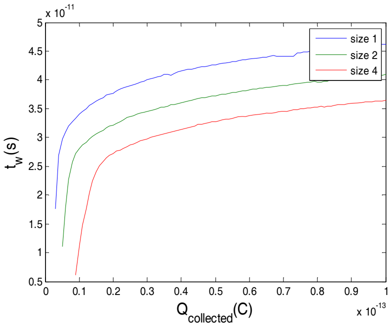

Figure 1. $Q_{collected}$-t_w characteristic of inverters

2.2. Glitch Width Model

Equation (4) defines the relation between node voltage $V(t)$ and $I(t)$, where $I(t)$ is the injected current, R and C is the node resistance and capacitance respectively.

$$C \times \frac{dV(t)}{dt} + \frac{V(t)}{R} = I(t) \qquad (4)$$

Equation (1) and (4) lead to the solution of V(t):

$$\begin{aligned} V(t) = &\frac{I_{peak} \times \tau_\alpha \times R}{\tau_\alpha - RC}(e^{\frac{-t}{\tau_\alpha}} - e^{\frac{-t}{RC}}) \\ &- \frac{I_{peak} \times \tau_\beta \times R}{\tau_\beta - RC}(e^{\frac{-t}{\tau_\beta}} - e^{\frac{-t}{RC}}) \end{aligned} \qquad (5)$$

But equation (5) does not hold true forever. Actually, the simulation result of a real circuit is illustrated in Figure 2. At each run of simulation, the I_{peak} of equation (1) is increased. The plot of glitches of each run is illustrated.

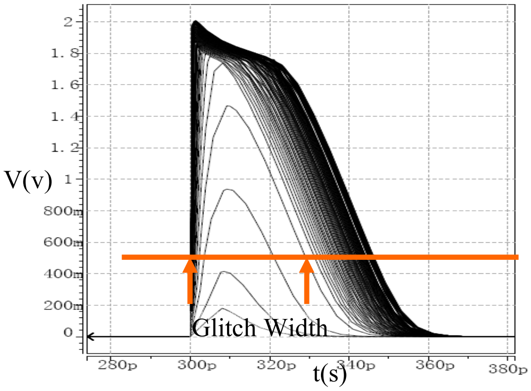

Figure 2. Simulation results of glitches

When the peak voltage approaches a maximum value, it refuses to rise any more. If we assume that the maximum peak voltage does not vary, then equation (5) leads us to the following conclusions:

1. $Q_{collected}$ corresponding to glitch width which is a little larger than the glitch width at the inflexion point will be huge.
2. Maximum glitch width scales down with technology scaling because RC scales down.

2.3. Transient Pulse Propagation Model

The research of transient pulse propagation shows that glitch degradation is mainly determined by propagation delay (t_p) of a logic cell [11].

For a logic cell propagation delay t_p and input voltage pulse duration τ_n, the glitch duration τ_{n+1} at the output of the cell can be estimated by (6), which is obtained by HSPICE simulation. When τ_n is smaller than t_p, the glitch can not propagate to the next stage.

$$\tau_{n+1} = 0.8469 \times t_p \times (\frac{\tau_n}{t_p} - e^{3.026\times(1-\frac{\tau_n}{t_p})}) \qquad (6)$$

2. Output Remapping

In this section, our output remapping technology is first proposed. We then explain that if static logic is not fast enough, we can use complex dynamic logic to replace the multi-stage logic at the output.

3.1. Output Remapping

First, glitches need to be propagated to the combinational output to cause a soft error. Second, according to section 2.3, if we increase the propagation delay of a gate, all the glitches narrower than its propagation delay can not pass it. Third, the density of the pulse width distribution drops very fast. In other words, the narrow glitches determine *SER*. Because $Q_{collected}$ increases significantly when t_w increases a little after the inflexion (Figure 1) and *SER* has an exponential relation with $Q_{collected}$ (equation (2)). Therefore we can replace some gates with another gate that have longer delay, so that narrow glitches will be filtered out at the output. If the static logic is not fast enough, we have to choose dynamic logic. In section 3.2, we will show the impact of stack depth on complex dynamic logic delay.

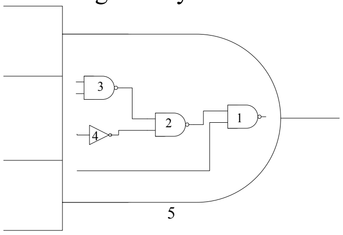

Figure 3. C2670 example

Example: Figure 3 is an example comes from ISCAS85 benchmark circuits (C2670) [13]. Gate 1 is connected to a critical output. Gate 1-4 is used to drive the output as a result of logic synthesis. Glitches are only filtered by the propagation delay of one NAND. If we replace gate 1-4 with a complex logic gate 5 that have longer delay, then glitches are filtered by the propagation delay of gate 5.

We use logical effort [12] to analyze as follows. The intrinsic delay of the static version of gate 5 is $(3+2+4+4)/3=13/3$. This comes from a NMOS gate of size 3, one PMOS gate of size 2 and two PMOS gates of size 4. The logical effort is $(4+3)/3=7/3$. So the propagation delay of a static gate 5 is $t_{inv}(13+7/3F)$, where F is the *path effective fan-out* which is defined as the load capacitance over the input capacitance. We can calculate the delay for dynamic and multi-stage version as well. The result is listed in Table 1 and illustrated in Figure 4. As you can see, the dynamic version is faster than multi-stage version while F is smaller than 10.

Static complex logic is slower than the multistage version in some certain circuits. For example, the delay of the 8NAND and 8NOR static gates is much larger than the multistage logic when F increases. So we have to choose dynamic logic for those.

Table 1. Logical effort analysis

static	Dynamic	multi-stage
$13/3+7/3F$	$5/3+4/3F$	6+4(F^(1/3))

In Figure 3, propagation delay from gate 3 input to gate 1 output is 34.8*ps*. We can adjust the load capacitance of gate 5 until its propagation delay approaches this value. Then all glitches narrower than 34.8*ps* can not propagate to the latch input or the output node. Refer to Figure 2 for the glitch generation model of inverters, and glitch width of 34.8*ps* corresponds to $Q_{collected}$ of 11.6*fC*. Due to the exponential relation between *SER* and $Q_{collected}$, *SER* is reduced significantly. Notice that all the other gates belong to the fan-in cone of this output remain the same and no delay penalty is introduced.

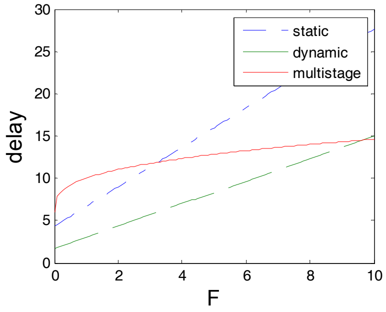

Figure 4. Delay comparison

3.2. Stack Depth Impact

Consider the dynamic logic with extreme conditions: 8 inputs. The relation between stack depth and maximum delay is listed in Table 2. We notice that NOR is much faster than NAND because the stack depth of NOR logic is much smaller. The observation leads us to this conclusion that the stack depth is the most important parameter which affects the delay. Our tests for the ISCAS85 benchmark circuits show that most of the critical outputs can be replaced with gates which have stack depth smaller than 4.

Table 2. Dynamic logic delay and stack depth

Stack depth	Maximum delay
1 (8NOR)	(17+2F)/3
2	(13+3F)/3
3	(9+4F)/3
4	(11+5F)/3
5	(7+6F)/3
6	(8+7F)/3
7	(9+8F)/3
8 (8NAND)	(10+9F)/3

4. Experimental Results

We have conducted tests on some benchmarks from ISCAS85. The logic cells used in our experiments is based on PTM 45nm models [14]. Each benchmark is synthesized with ABC [15], and then the output is remapped. The stack depth of the remapped gate is calculated simultaneously.

The results are demonstrated in Table 3 as follows. Take C2670 for example, original t_p in the table is the propagation delay of the gate 4 in Figure 3; optimized t_p is the t_p of gate 5. $Q_{critical}$ in the table refers to glitch generation model of size 1 inverter, because it is the most vulnerable cell to soft error. After the remapping, the glitch needs to be wider than *34.8ps* to propagate to the latch or the output node of the circuit. According to our glitch generation model, this glitch width leads to significantly rise of $Q_{critical}$. Furthermore, only one remapped output gate in our tests needs to be implemented with dynamic logic other than the static logic. If the static logic is not replaced by dynamic logic, the delay penalty will be 4%.

As we mentioned before, glitch width scales down with technology scaling, so we expect this method scales well. The simulation results of relation between glitch width and gate delay at different technology nodes are presented in Table 4. In the table, t_w/t_p is the glitch width over propagation delay. Both are simulation results of inverters. The current source is connected to the inverter output. The load is two inverters of the same size.

Our method only needs a little change at the combinational output, so the power/area penalty is limited. It can be used on critical path and no delay penalty is introduced.

5. Conclusions

In this paper, an output remapping technique is proposed to reduce *SER*. Experimental results shows up to *20X* increase in $Q_{critical}$. Because the area of the remapped gates is small compared to the circuit and the output node capacitance is large, *SER* caused by those gates can be ignored. Our analysis also shows that glitch width scales down with technology scaling, so we expect this method scales well. The power/area penalty is limited and there is no delay penalty. We have to use dynamic logic when the remapped static logic is not fast enough. This causes difficulty in implementation of our output remapping technique, however, the dynamic logic is rarely used (only one) during our tests.

References

[1] P. Hazucha and C. Svensson, "Impact of CMOS technology scaling on the atmospheric neutron soft error rate," IEEE Trans. Nucl. Sci., Vol. 47, No. 6, pp. 2586-2594, 2000.

[2] P. Shivakumar, M. Kistler, S. Keckler, D. Burger, and L. Alvisi, "Modeling the effect of technology trends on the soft error rate of combinational logic," ICDSN, pp. 389-398, 2002.

[3] N. Seifert, X. Zhu, and L.W. Massengill, "Impact of scaling on soft-error rates in commercial microprocessors," IEEE Trans. Nucl. Sci., Vol. 49, pp. 3100-3106, 2002.

[4] D. P. Siewiorek and R. S. Swarz, "Reliable Computer Systems: Design and Evaluation (3rd edition)," A. K. Peters, 1998.

[5] M. Nicolaidis,"Time redundancy based soft-error tolerance to rescue nanometer technologies," Proc. VTS, 1999, pp. 86–94.

[6] K. Mohanram and N. A. Touba, "Cost-Effective approach for Reducing Soft Error Failure Rate in Logic Circuits," Proc. Int'l Test Conference (ITC), pp 893-901, Sep. 2003.

[7] Q. Zhou and K. Mohanram, "Gate Sizing to Radiation Harden Combinational Logic," IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, Vol. 25, No. 1, Jan. 2006.

[8] Y. S. Dhillon, A. U. Diril, A. Chatterjee and A. D. Singh, "Analysis and optimization of nanometer CMOS circuits for soft-error tolerance," IEEE Transactions on Very Large Scale Integration (VLSI) Systems, pp 514-524, 2006.

[9] V. Joshi, R. R. Rao, D. Blaauw, and D. Sylvester, "Logic SER Reduction through Flipflop Redesign," Proc. Int'l Symposium on Quality Electronic Design (ISQED), pp. 611-616, 2006.

[10] Q.Zhou and K.Mohanram, "Cost-effective radiation hardening technique for logic circuits", ICCAD, pp.100-106, 2004.

[11] Y. S. Dhillon, A. U. Diril and A. Chatterjee, "Soft-error tolerance analysis and optimization of nanometer circuits," DATE, pp 288-293, 2005.

[12] B. Hu, Y. Watanabe and M. Marek-Sadowska. "Gain-Based Technology Mapping for Discrete-Size Cell Libraries." DAC, pp. 574–579, 2003.

[13] F. Brglez, H. Fujiwara, "A neural netlist of ten combinational benchmark circuits and translator in Fortran," Intl. Symp. on Circuits and Systems (ISCAS), pp. 663-698, Jun 1985.

[14] Y. Cao, T. Sato, D. Sylvester, M. Orshansky and C. Hu, "New paradigm of predictive MOSFET and interconnect modeling for early circuit design," pp. 201-204, CICC, 2000.

[15] Berkeley Logic Synthesis and Verification Group, "ABC: A System for Sequential Synthesis and Verification, Release 51205." http://www.eecs.berkeley.edu/~alanmi/abc/.

Table 4 Glitch width scales down with gate delay

	180nm	90nm	65nm	45nm
t_w/t_p	4.2	6.2	5.3	4.9

Table 3 Glitch-filtering effect

Benchmar	Original t_p	Original $Q_{critical}$	Optimized t_p	Optimized $Q_{critical}$	Stack depth
C432	*11.6ps*	*<1fC*	*32.5 ps*	*7.7fC*	*3*
C499	*21.2ps*	*3.4fC*	*44.4ps*	*70.1fC*	*3*
C1908	*21.2ps*	*3.4fC*	*44.4ps*	*70.1fC*	*2*
C2670	*11.6ps*	*<1fC*	*34.8ps*	*11.6fC*	*3*
C3540	*11.6ps*	*<1fC*	*32.5ps*	*7.7fC*	*2*
C5315	*11.6ps*	*<1fC*	*32.5ps*	*7.7fC*	*2*
C7552	*11.6ps*	*<1fC*	*32.5ps*	*7.7fC*	*2*