Crosstalk Noise in FPGAs

Yajun Ran and Malgorzata Marek-Sadowska
Department of Electrical and Computer Engineering
University of California, Santa Barbara
Santa Barbara, CA 93106
{ranyj, mms}@ece.ucsb.edu

ABSTRACT

In recent years, due to rapid advances in VLSI manufacturing technology capable of packing more and more devices and wires on a chip, crosstalk has emerged as a serious problem affecting circuit reliability. Even though FPGAs are more immune to crosstalk noise than their ASIC counterparts manufactured in the same technological process, we have reached the point where FPGAs have become affected by crosstalk as well. Because FPGAs have regular interconnect structures, crosstalk noise can be more easily controlled. In this paper, we investigate the crosstalk noise in FPGAs and propose new strategies to reduce its impact on delay. Our methods can reduce crosstalk noise by statistically significant amounts with no penalty in performance, power, or area.

Categories and Subject Descriptors

B.7.1 [**Integrated Circuits**]: Types and Design Styles

General Terms

Design

Keywords

Crosstalk, noise, FPGAs, switch box

1. INTRODUCTION

As VLSI technology advances, feature sizes shrink and devices become smaller and faster. At the same time, wire resistance and capacitance decrease at a much slower rate or even proportionally increase, resulting in wire delay taking up an increasingly larger portion of a total path delay. On the other hand, the wires which connect devices become narrower and taller; as a result their coupling capacitance dominates the total interconnect capacitance. Coupling capacitance may cause neighbor switching wires (*aggressors*) to introduce parasitic noise pulses on their quiet neighbors (*victims*), possibly causing functional failure if the noise-induced incorrect logic value is latched. Moreover, simultaneously switching neighboring wires experience speed-up or slow-down due to crosstalk, which increases their delay variations. The dominance of wire delay over gate delay and coupling capacitance over the capacitance to ground cause the delay variation due to neighboring wires switching increasingly significant. Because of difficulties in predicting the switching activity on neighboring wires, the slowdown and speedup effects induced by capacitive coupling make the critical path delay estimation intractable. The common design practice of satisfying the specifications under the worst case scenario is very pessimistic because it assumes that all the noise can occur simultaneously and be accumulated. This practice takes more design effort than necessary, resulting in longer time-to-market and higher overall costs.

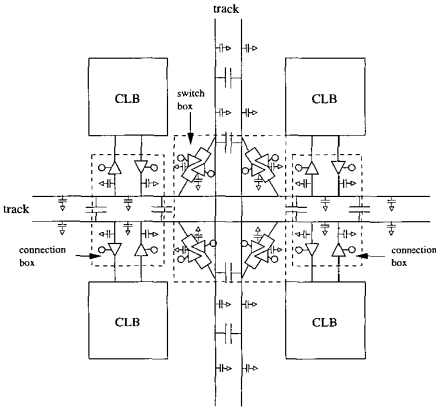


Figure 1. Capacitance in FPGAs.

In recent years the field-programmable gate arrays (FPGAs) have become one of the most popular design fabrics because of their fast time-to-market, low nonrecurring engineering costs, and easy debugging. A typical FPGA structure, such as island-style FPGA, is composed of configurable logic blocks (CLBs), I/O blocks and programmable routing. Each CLB consists of one or more basic logic elements (BLEs) each of which contains a lookup table (LUT) and a flip-flop. Interconnect programmability is implemented by switches inside connection boxes and switch boxes. Connection boxes allow CLB pins to connect to tracks. Switch boxes are used to build connections of appropriate lengths from prefabricated wire segments along the tracks. Appropriate programming of the switches in a connection box allows CLB to tap into a selected track. Programming the switch boxes permits building interconnects between different CLBs. FPGAs began to encounter crosstalk noise problems later than ASICs. Compared to ASICs, FPGAs have some good features which make them somewhat immune

DAC 2003, June 2-6, 2003, Anaheim, California, USA.

to crosstalk noise. An FPGA chip contains many buffers (switches) of the same size, so that strong-aggressor, weak-victim combinations seldom occur. Because of the relatively short distances between consecutive switches, slew rates do not degrade as much. Furthermore, there are many capacitance sources along a track, such as the wire capacitance itself and the input and output capacitances of switches (connecting to CLB pin or another wire segment). Figure 1 shows the capacitances in FPGA. Due to those capacitances, the coupling capacitance is a smaller fraction of the total capacitance. In other words, those capacitances which make a circuit slower contribute to stabilizing the path delay by reducing the impact of neighboring switching. But with the process shrinking, the crosstalk noise problem has become increasingly serious in FPGAs.

To the best of our knowledge, there are only a few papers addressing this problem. In [9], a routing algorithm has been modified to take capacitive coupling lengths into account for the purpose of reducing crosstalk-induced slowdown on a critical path. With regular structures in FPGA, there are more ways to reduce crosstalk noise. Here, we systematically investigate crosstalk effects in FPGAs and propose new strategies to reduce crosstalk effects. In FPGAs, functional glitch noise is not a big problem since all the latches are well buffered; therefore we are focusing here on crosstalk-induced delay jitter.

This paper is organized as follows. In section 2 we describe the FPGA architecture used for experiments, and in section 3 we investigate the crosstalk's impact on FPGA with the technology advancing. In section 4, we examine the strategies to reduce crosstalk effects. Section 5 concludes the paper.

2. EXPERIMENTAL ARCHITECTURE

There are many factors to be considered while designing an FPGA architecture. Among them, switch box topology, track segmentation scheme, cluster size and switch size are the most important. The flexibility of interconnects in FPGAs is measured by two parameters, F_s and F_c. F_s is defined as the total number of possible connections each wire entering a switch box can make. F_c is defined as the number of tracks to which each logical pin can connect [6]. Usually a track is segmented for better performance. Different segmentation schemes could be used for different tracks.

Focusing on crosstalk noise in FPGA, we choose a typical FPGA architecture for our experiments because exploring all the possibilities would be impossible. But our conclusions should apply to most of the architectures. We consider a cluster-based symmetrical island-style FPGA architecture. Each CLB contains four 4-LUTs and four FFs and has 10 inputs and 4 outputs. The switch box is a classic subset switch box, except when explained explicitly, and F_s is 3. F_c is $0.5N$ where N is the number of tracks. We use two possible segmentation schemes, length 1 and length 4, in terms of logic block dimensions. In each scheme, all the tracks have the same segmentation.

3. CROSSTALK NOISE IN FPGA

A path delay in FPGA can be divided into two parts, a LUT delay and an interconnect delay, which in turn includes wire delay and the delay due to switch buffers. A LUT is composed of memory units with auxiliary circuitry. Because of their ability to implement any functions, LUTs are much slower than the gate implementations with fewer levels in ASIC. But FPGAs still have the same trend of delay with process shrinking as ASIC. Figure 2 shows the trend of the ratio of interconnect delay versus logic delay. The experiments are done as follows. First, we run Spice simulation to get the delay of one logic stage (through a LUT and input multiplexers). Second, we run VPR [3] (a placement and routing tool for FPGA) to get the average interconnect length between two logic stages in a critical path of the 20 largest MCNC benchmarks. We use single-length segments here for the convenience of counting the interconnect length. The average number we get is 11 block lengths. After that, we calculate all the resistance and capacitance (both wire capacitance and switch capacitance) in one single-length segment using the Berkeley predictive technology model [1]. From Spice simulation, we get the delay through one segment. The driving buffer size is 6X minimum size [2]. We multiply the average interconnect length (in terms of logic block dimensions) by the delay of one single-length wire, and we get the wire delay in Figure 2. Figure 2 also shows the ratio of coupling capacitance over total capacitance assuming there are two adjacent segments. For all the experiments, we use minimum space between segments. Because interconnect delay will contribute to a larger and larger portion of the total path delay, and capacitive coupling will take an increasingly larger portion of the total capacitance, it is obvious that the delay variation (either slowdown or speedup) induced by crosstalk will become more of a challenge in FPGAs.

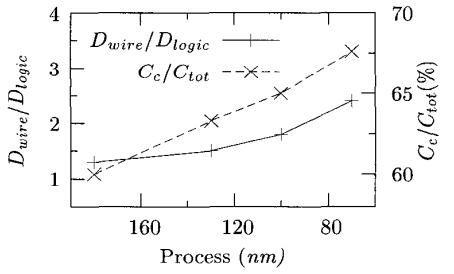


Figure 2. Trends of delay and coupling capacitance

4. REDUCING CROSSTALK EFFECTS

The main cause of the possible noise is the coupling capacitance coming from the physical adjacency between wires. But there are some strict conditions for crosstalk noise to become a real problem. First of all, an aggressor must switch at a specific time window with respect to the transition of the victim in order to have a significant effect on a victim's transition. This is called temporal correlation. In addition, an aggressor must switch in the opposite (for slowdown effect) or the same (for speedup effect) direction to the victim's transition. This is called functional correlation. Other factors including aggressor's and victim's driving strengths as well as their slew rates will also affect crosstalk-induced delay variation. The common reduction strategies are targeting to eliminate or reduce some of these conditions in which noise can occur. In this section, we will investigate some of the existing methods and propose two new strategies to reduce crosstalk noise impact in FPGA.

4.1 Wire spacing

The coupling capacitance will be dramatically reduced if we increase the spacing between adjacent wires. The total loading capacitance will also be reduced by wire spacing; hence the circuit timing and power consumption can be improved as well. Figure 3 shows the ratio of coupling capacitance over the total capacitance for different wire spacings. We can see the coupling capacitance is reduced drastically with the space increase. The reduction of coupling capacitance will be directly converted into the reduction of delay variation due to crosstalk. The effectiveness of wire spacing is decided by available routing space. If the total area of an FPGA structure is decided by the transistors, and the extra spacing between wires does not cause extra area penalty, wire spacing is very effective. For those architectures in which wires decide the total area, wire spacing may be unaffordable. In [7], with careful layout, 49 tracks can be packed above a switch box with minimum width and minimum spacing. Considering that the routing tracks needed for a cluster-based FPGA architecture can range from 20 to 50, increasing the spacing between all the wires is not practical, so some wire spacings might be favored for use for fast-path routing as long as there is not too much area penalty. In [2], a 13% delay improvement is reported for 20% of total tracks using 5X minimum wire spacing in $0.35\mu m$ technology. So the trade-off between area and performance should be considered carefully.

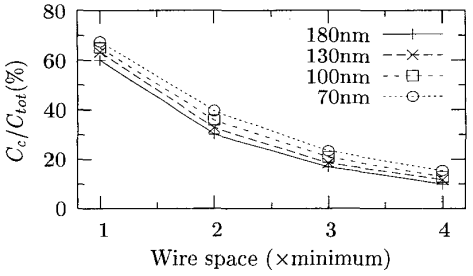


Figure 3. Effects of wire spacing

4.2 Shielding

Shielding is another technique commonly used. Inserting a VDD or GND track between two signal wires can eliminate neighboring switching explicitly. The coupling capacitance is not reduced in this case, so shielding is not as effective in improving the timing as wire spacing with the same area overhead [5]. But the advantage of shielding is that it provides a good termination thus leading to better delay predictability. It introduces more confidence to noise control. This technique has been used extensively for the long lines crossing the entire chip and for some dedicated global interconnects such as clock/enable nets. Those interconnects suffer more from noise due to their long runs. For common signal nets, wire spacing might be better than shielding.

We can also make use of empty tracks as shields. In [9] the author reports a crosstalk-aware router which tries to separate critical nets from others by utilizing unused segments. The improvement is limited by the router's capability and the number of available segments as well as the routability of the architecture.

4.3 Buffer sizing

By sizing up the switch buffers, we can get stronger driving capabilities and reduce the interconnect delay, and also reduce the delay variation by crosstalk. Due to the faster transition, a wire will suffer less from crosstalk. It would be impractical to size up all the switch buffers in FPGAs. It would be more realistic to size up a fraction, say 20% of buffers, and let the router use those stronger buffers for critical path routing. Because the strong buffers might become strong aggressors for unsized buffers, some shields or extra spacing will be needed between these two groups. The penalty for buffer sizing is greater area cost and more power consumption.

For that reason, we prefer switch buffers instead of pass transistors. Switch buffers can minimize the impact of fanout and also avoid the slew rate degradation, which are helpful in reducing crosstalk noise. Xilinx has used fully-buffered interconnections in their Virtex-II architecture [10]. In our experiments, all the switches are buffered.

4.4 Noise distribution

Because of the temporal and functional correlation, it is unlikely that all the slowdown along a critical path can be aggregated. So if two critical paths have the same worst-case delay variation (simple summation of the maximum slowdown of all stages), but different numbers of aggressors, we prefer the more distributed one with more aggressors because each aggressor causes less noise. Below we explain our reasoning.

4.4.1 Statistical analysis

Let random variable X_i represent the probability of a stage i having noise of x (here we concentrate on slowdown, since speedup is similar). Figure 4 shows a typical curve of slowdown. The horizontal axis shows the difference between the aggressor's switching time and the victim's switching time. We observe that when the aggressor switches at a specific time window with respect to the victim's switching time, there is a significant slowdown effect. This range usually is approximately equal to the sum of the victim's and aggressor's slew rates. The approximated worst case slowdown can happen only within a very narrow window. When it happens, we say that the aggressor and victim are aligned. Achieving the perfect alignment to get the worst case slowdown is difficult (or the probability to get worst case noise is very small), we use an exponential distribution $f(x)$ as an approximated probability density function for X_i.

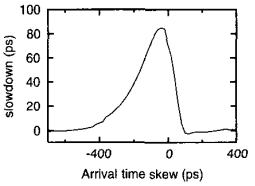


Figure 4. Slowdown vs. arrival time skew

$$f(x) = \frac{e^{-\frac{x}{\alpha}}}{\alpha} \quad (1)$$

The probability $P(X > x)$ of slowdown larger than x can be calculated from the accumulated distribution function $F(x)$.

$$P(X > x) = 1 - F(x) = e^{-\frac{x}{\alpha}} \quad (2)$$

In general, the range of a random variable is infinite in the probability density function. We approximate the worst-case

slowdown by assuming that $P_0 = P(X > x_0)$ is the probability of the worst-case noise occurring, and x_0 is the worst-case noise value we get from noise calculation. Hence, we can derive α from P_0.

$$\alpha = -\frac{x_0}{\ln P_0} \tag{3}$$

Suppose a path has n stages with possible slowdown effects. We can determine the probability density function of the accumulated slowdown along the path. For simplicity of estimation, we assume that all stages are independent and have the same slowdown distribution. Therefore we get the probability density function of the summation Y, a gamma distribution.

$$f(y) = \frac{y^{n-1} e^{-\frac{y}{\alpha}}}{\alpha^n \Gamma(n)}, \qquad Y = \sum_{i=1}^{n} X_i \tag{4}$$

When n is large (which is typical since the number of stages in a path usually exceeds 10), the gamma distribution can be approximated by normal distribution with mean $n\alpha$ and variance $n\alpha^2$. Figure 5 shows an example where Case 2 has twice the number of stages as Case 1 ($n_2 = 2n_1$), but each stage has half the worst-case slowdown of another ($x_{2,0} = x_{1,0}/2$). From Eq. 3, we have $\alpha_2 = \alpha_1/2$. The resulting two normal distribution functions of the accumulated worst-case slowdown have the same mean $n_1\alpha_1 (= n_2\alpha_2)$, but different variances. $\sigma_2 = n_2\alpha_2^2 = \sigma_1/2$. After normalization, Case 1 has variance σ of 2 and Case 2 has variance σ of 1 in Figure 5. If we count the probability (the area under the curve from the dashed vertical line to ∞) of worst-case accumulated slowdown $n_1 x_{1,0} (= n_2 x_{2,0})$, Case 2 (solid curve) will have a much smaller probability than the Case 1 (dashed curve).

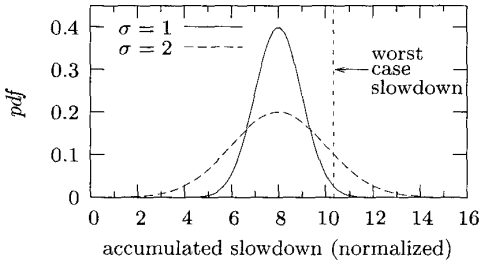


Figure 5. Worst-case accumulated path delay noise

Based on this analysis, we believe more distributed noise sources will help to prevent worst-case noise accumulation by dramatically increasing the difficulty of worst-case alignments.

4.4.2 New switch box design

According to the above analysis, we would like to avoid long parallel wires in which crosstalk-induced delay can be more easily accumulated. In FPGA all tracks have similar structure. Once two adjacent segments have overlapped switching windows, they will most likely remain overlapped along the whole coupling length, making the crosstalk effects worse. This analysis motivates a new switch box structure which permutes the connections and breaks the possible long parallel wires, making two wires adjacent on a distance no longer than the length between two consecutive switch boxes.

Many papers have been published on switch box design for better performance and more compact area. The classic *subset* switch box used in Xilinx XC4000-series (Figure 6(a), also called *disjoint*) is area-effective but creates disjoint routing domains. The *universal* switch box (Figure 6(b)) proposed in [4] can simultaneously route all two-point connections in the switch box. The *Wilton* switch box [8] (Figure 6(c)) eliminates routing domains by rotating tracks by one to create connections between different domains and provide greater routing flexibility. Previously switch boxes have not paid any particular attention to the adjacency relationships of the segments. If the physical order of the tracks is the same as their logical order (from 1 to N), all of the previous switch boxes have the common property that a segment has the same neighbors before and after a switch. This is not good from the crosstalk noise point of view. If we can make a switch box which can change the neighbors such that two original neighbors are separated after a switch, then all the long parallel running wires will be broken by the switch boxes, and the maximum coupling length between two wires can be at most one segment. An example proposed by us, *twist* switch box, is shown in Figure 6(d). It is easy to observe that any two adjacent signals will no longer be adjacent after passing through the switch box.

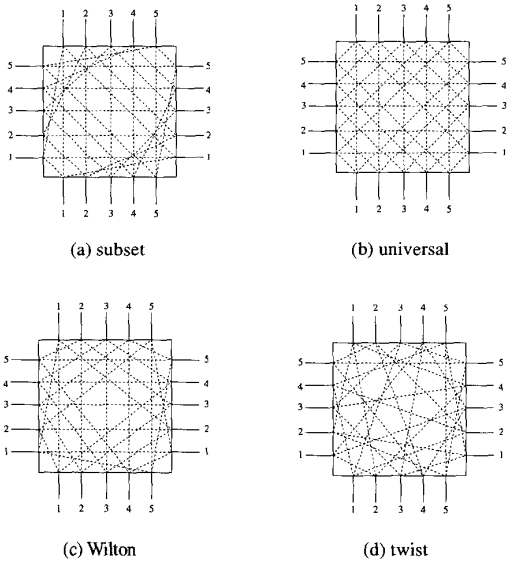


Figure 6. Different switch boxes

The key of this design is to find a suitable mapping function which will implement this transformation. Given a switch box with four sides, each side consisting of N tracks, there are $6N$ switches in a switch box (assuming each switch to be bidirectional). Figure 7 shows the possible six connection types. Table 1 shows the mapping functions used in different switch boxes. By introducing a separation factor *SPF* which determines the distance of two originally adjacent wires after the switch and multiplying it by the track number in the mapping function of *Wilton* switch box, we can get the *twist* switch box. The *Wilton* switch box can be thought of as a special example with *SPF* 1.

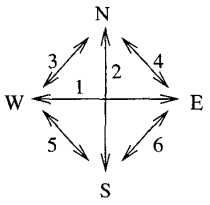


Figure 7. Connection types in a switch box

There are many choices for the separation factor; it can be arbitrary as long as it is larger than 1, less than N-1 and is not a divisor of N for unique mapping. It can be proved that a proper *SPF* always exists for N larger than 4 (the proof is omitted here due to page limit).

The routability of *twist* switch box is similar to that of *Wilton* switch box since it also creates connections between different routing domains. But due to the skewed connection of this new switch box, more routing resources are needed. In the new switch box, track W_i may connect to a track $E_j (i \neq j)$, so it requires additional wire connections which will cross other wires in the same direction. Hence it needs wire segments on additional routing layer to build those connections. When crosstalk noise becomes more of a problem, the cost of extra routing resources may be worthwhile expense.

Table 1. Mapping functions for different switch boxes

conn type	sub set	univ.	Wilton	twist
$f_{WE}(t)$	t	t	t	$SPF \times t$
$f_{SN}(t)$	t	t	t	$SPF \times t$
$f_{WN}(t)$	t	$N-t+1$	$N-t+2$	$SPF \times (N-t)+2$
$f_{NE}(t)$	t	t	$t+1$	$SPF \times t+1$
$f_{WS}(t)$	t	t	$t-1$	$SPF \times t-1$
$f_{SE}(t)$	t	$N-t+1$	$N-t$	$SPF \times (N-t)$

4.4.3 Experimental results

The improvements due to this new switch box depend on how prevalent long parallel interconnect runs become in conventional architectures. Table 2 shows the experimental results for the 20 largest MCNC benchmark circuits with two segmentation schemes. We use VPR [3] to do the placement and routing. For each circuit, we first determine the minimum number of tracks needed to complete the routing and then redo the routing for 20% extra tracks to emulate the "low stress" conditions which occur in practice. After the modification of VPR's timing analysis part, we consider the effects of crosstalk-induced delay noise. For crosstalk-induced delay variation, we use the simple $2C_c$ model, which treats an aggressor with opposite switching direction to victim as a connected-to-ground capacitor of $2C_c$. In Table 2, #Agg is the average number of aggressors on the critical path for those circuits, L_c is the average coupling length of each aggressor, and L_{mc} is the average maximum coupling length in each critical path. Coupling lengths are expressed in terms of logic blocks. When counting aggressors, if two continuing aggressing segments are adjacent to the victim net, we consider them as belonging to the same aggressor even though they might be separated by some switches.

For the four different switch box configurations, the worst-case critical path delays and worst-case slowdown of the path

Table 2. Coupling situation for different switch boxes

switch box	segment length 1			segment length 4		
	#Agg	L_c	L_{mc}	#Agg	L_c	L_{mc}
subset	50.3	2.1	9.0	29.6	3.8	11.2
univ.	51.3	2.1	8.7	28.2	3.9	12.5
Wilton	51.5	2.1	8.8	27.3	3.9	12.0
twist	103.1	1.0	1.0	42.0	2.6	3.5

are similar (less than 3%). The average delay uncertainty of the critical path, which is the worst-case slowdown over the nominal case path delay, is about 15% for single-length segments and about 25% for length 4 segments. For single-length segment architecture, the coupling length of each aggressor is always 1 for the *twist* switch box since each segment has been twisted such that the next stage will have different tracks as neighbors. And the average coupling lengths of all the other switch boxes are more than 2. Given similar worst accumulated slowdown effects, the worst alignment scenario will be more likely to happen for all the other switch boxes based on the previous analysis. In contrast, with this new switch box, all the aggressors provide only one segment of coupling, considering the temporal and functional correlation. It is much more improbable for them to be aligned to produce the worst-case slowdown. For length 4 segmentation architecture, the average aggressor coupling length of other switch boxes is still about 50% greater than for the *twist* switch box. When we look at the longest parallel running segments on a critical path (L_{mc}), we see that all the previous switch boxes may have caused long coupled segments, which are conducive to crosstalk noise. In one design, the longest coupling length stretches the length of 34 blocks, which is dangerous for circuit timing. But for the *twist* switch box, the longest coupling length is always one segment.

4.5 Cancellation of slowdown and speedup

In previous works, the crosstalk-induced slowdown and speedup effects are considered separately, which means the switching windows are always expanding at both ends. In general situations, it is hard to determine the correlation between slowdown and speedup, but for some regular structures, like FPGA here, we might be able to use speedup to cancel out some slowdown effects. This idea was originally proposed for buses in [5]. The experiments have shown that by shifting the inverter locations along a set of long lines, the worst delay can be reduced up to 30%.

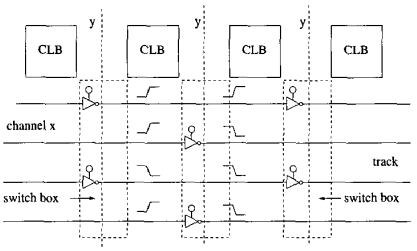
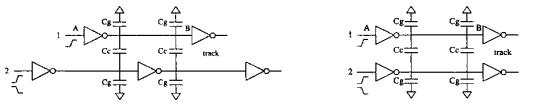


Figure 8. switch inverters to reduce crosstalk effects

In FPGA, the tracks are regularly segmented wires running in parallel. The staggering of switch buffers will help to skew the

switching windows of neighboring wires. But the existing switch buffers do not change the polarity of signals. In fact, if we use inverters as switch buffers and interleave the tracks as shown in Figure 8, we can cancel out some delay noise once two adjacent wires run in parallel at least the length of one segment. Because of the skewed locations of the inverters, two signals will have the same transition at half of the segments and the opposite transition at another half, which means a signal will be slowed down for a half-length and sped up for another half-length. This technique is only effective when segment length is larger than 1 (segment length is 2 in Figure 8). It requires that switches are shifted by half of the segment length with respect to the adjacent segments.



(a) skewed inverter (b) aligned inverter

Figure 9. Skewed vs. aligned switch inverter

An experiment simulating a two-track system shows an improvement of delay uncertainty made possible by this scheme. Figure 9 illustrates the two structures we compare, the skewed inverter and aligned inverter distribution. The inverter size is 5X minimum, the wire length is $400\mu m$ between the two inverters, C_g is $6fF/100\mu m$, and C_c is $8.7fF/100\mu m$ in $0.13\mu m$ technology. Line 1 is always falling at a fixed time point. We are observing the delay from A to B when line 2 is rising or falling, with arrival time changing from 700ps before line 1's transition to 400ps after line 1's transition. Figure 10 shows the delay due to the attacking line 2, where S/A stands for skewed inverter and aligned inverter, respectively, and R/F stands for line 2 rising and falling, respectively. For aligned inverter structure, the delay variation can amount to 70%. With the skewed inverter structure and the cancellation effects, the variation peak has been reduced to less than 30% in the worst case, regardless of the aggressors' switching direction.

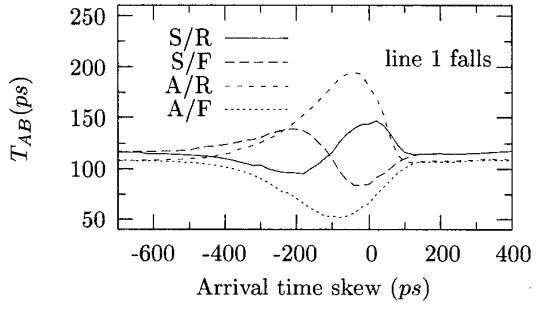


Figure 10. Crosstalk noise by inverter relocations

Using inverters as switches might cause an incorrect signal polarity at the LUT inputs. However, LUT itself can implement any function, so we need only to modify the programming configurations to correct this mismatch. Suppose a LUT implements a function $f(x_1, ..., x_n)$ and x_i is inverted. To correct the mismatch, we can swap the contents of the lookup table entries due to x_i and those corresponding entries due to $\bar{x}_i$.

Replacing switch buffers with inverters does not cause any penalty and extra implementation efforts, and the parallel runs of segments will benefit from this configuration. Compared to the *twist* switch box method, inverters do not change routing of the previous architectures and only require that the switch inverters are interleaved by half of the segment.

5. CONCLUSIONS

As VLSI processes shrink, crosstalk noise becomes more serious even in FPGAs. In this paper, we have investigated some existing methods to reduce crosstalk effects. Wire spacing and shielding are very effective, but their effectiveness depends on the available routing resources. Some wire spacing and buffer sizing could be used for critical path routing, but it requires an intelligent router to make good use of them. We propose a *twist* switch box which breaks the possible parallel running wires and makes worst-case alignment more unlikely to occur. It does not need any extra effort on a router but does need more routing resources to build the cross connections. The effectiveness of *twist* switch box can be examined using a subtle timing analysis tool that takes into consideration the statistical nature of crosstalk noise. In addition, we suggest using switch inverters instead of switch buffers to cancel out some slowdown effects by using speedup effects. By reducing the delay variation, we achieve better predictability. This modification is suitable for segmented routing architecture and does not incur any overhead. The delay noise is naturally suppressed by the cancellation for parallel running segments.

6. ACKNOWLEDGEMENTS

This work was supported in part by NSF grant 0098069 and by the State of California MICRO Program through Xilinx and Mentor Graphics.

7. REFERENCES

[1] Berkeley predictive technology model. http://www-device.eecs.berkeley.edu/~ptm/introduction.html.

[2] Betz, V. and Rose, J., "Circuit Design, Transistor Sizing and Wire Layout of FPGA Interconnect", *CICC'99*, pp. 171–174, 1999.

[3] Betz, V. and Rose, J., "VPR: A New Packing, Placement and Routing Tool for FPGA Research", *FPGA'97*, pp. 213–222, 1997.

[4] Chang, Y.-W., Wong, D. and Wong, C., "Universal Switch Modules for FPGA Design", *ACM TDAES*, pp. 80–101, January 1996.

[5] Kahng, A., Muddu, S., Sarto,E. and Sharma, R., "Interconnect Tuning Strategies for High-Performance ICs", *DATE'98*, pp. 471–478, 1998.

[6] Rose, J. and Brown, S., "Flexibility of Interconnection Structures for Field-Programmable Gate Arrays", *IEEE JSSC*, pp. 277-282, March 1991.

[7] Schmit, H. and Chandra, V., "FPGA Switch Block Layout and Evaluation", *FPGA'02*, pp. 11–18, 2002.

[8] Wilton, S. *Architecture and Algorithms for Field-Programmable Gate Arrays with Embedded Memory*. PhD thesis, University of Toronto, 1997

[9] Wilton, S., "A Crosstalk-Aware Timing-Driven Router for FPGAs", *FPGA'01*, pp. 21–28, 2001.

[10] Xilinx. *Online Databook*. San Jose, CA, 2002.