An Improved False Aggressor Pruning Algorithm for Effective Crosstalk Tests

Hyungjun Cho
Department of Electrical & Electronic Engineering
Yonsei University
Seoul, Korea
chj0937@soc.yonsei.ac.kr

Kyungmin Kim
Department of Electrical & Electronic Engineering
Yonsei University
Seoul, Korea
gamesety@ yosei.ac.kr

Youbean Kim
Department of Electrical & Electronic Engineering
Yonsei University
Seoul, Korea
inyacio@soc.yosei.ac.kr

Sungho Kang
Department of Electrical & Electronic Engineering
Yonsei University
Seoul, Korea
shkang@yonsei.ac.kr

Abstract - *In this paper, we present an efficient false aggressor pruning algorithm with functional correlation. A back-tracing algorithm and a logic implication approach are used to identify false coupling interaction between coupled lines. Crosstalk noises between neighboring wires can make timing errors or functional failures. The pessimism due to false aggressors must be minimized to reduce the additional design cost. The proposed algorithm increases the accuracy in static timing analysis by pruning false aggressors and decreases the pessimism efficiently. The proposed algorithm can reduce the time consumption of the crosstalk test by pruning the false aggressors. The proposed algorithm can increase the accuracy of crosstalk test by handling glitches and overshoots which are not considered in the previous works.*

Keywords: *crosstalk noise, aggressor, victim, glitch, overshoot.*

1 Introduction

The coupling capacitance of neighboring lines can have a significant impact on gate delay in today's deep submicron circuits. When two coupled lines switch in the opposite direction, the interconnect delay increases. If they switch in the same direction, the interconnect delay decreases. The additional buffer insertion or the space increase of adjacent lines is required to prevent the crosstalk noise. These increase the chip area and the power consumption. The crosstalk can cause not only delay variations but also functional failures [1].

In order to correct crosstalk noises, it is demanded additional design costs. Therefore, aggressor nodes should have to prune when it cannot affect to a victim node. In this paper, we propose an efficient false aggressor pruning algorithm with functional correlation. Path sensitizations and a logic implication approach are used to identify false coupling interaction between coupled lines.

The crosstalk noise analysis is essential for accurate static timing analysis (STA) because crosstalk noises can cause problems which affect the delay of the circuit. The pessimism due to false aggressors must be minimized to reduce the additional design costs.

The proposed algorithm decreases the pessimism efficiently and increases the accuracy in STA by pruning false aggressors. If a temporal pruning and a functional pruning are applied simultaneously, better results can be obtained. However it may suffer from severe computation time.

STA does not consider the functional correlations between nets. Therefore, the information from STA and the results of noise analysis from switching timing windows [2] might be overly pessimistic. The main goal of the proposed algorithm is to improve the accuracy of noise analysis by considering the functional correlations between neighboring nets. The proposed algorithm concerned about glitches and overshoots which are not considered in the previous works [3-5].

2 Crosstalk effects

Among all the noise effects, coupling effects between adjacent interconnects, namely crosstalk, has become a very important issue. Two types of coupling effects have been introduced: crosstalk-induced glitch and crosstalk-induced delay. Both effects can have a bad impact on circuit performance. The crosstalk between two lines that are nearly simultaneously switching in the same or the opposite direction may result in decrease or increase of delays on both lines compared to the delays when only one line is switching.

A variety of the crosstalk analysis model has been proposed in [4-6]. In general, the degree of crosstalk depends on several factors. Attributes such as drive strength (R_{ON}), line length, clock speed, skew, driver balance, load to load balance, and impedance matching all contribute to the degrees in which crosstalk can vary. Even if crosstalk noises are minimized during design,

process variations and defects during manufacturing may introduce excessive cross-coupling capacitance and inductance between interconnects, resulting in increased noise [4].

The adverse effects of increased cross-coupling capacitance and inductance on signal integrity can be threefold. When the cross-coupled capacitance becomes a first order parameter between two bus lines, two basic signal anomalies can take places as a result to step inputs. When one signal is switched and the other is driven steady the energy transfer through coupling capacitance (Cc) results in a voltage glitch on the steady signal. This is shown in Figure 1. This is called '*crosstalk glitch*'.

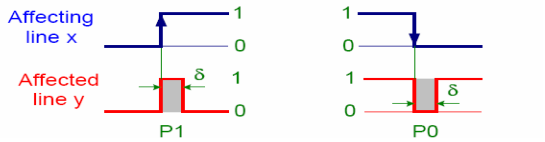

Figure 1. Crosstalk noises (glitch)

The second anomaly, when the two lines are switched to same transitions, the result is a decrease in transition time, as shown in Figure 2(a). This is called '*crosstalk speed-up*'. In contrast to speed-up, when the two lines are switched to opposite transitions, the result is an increase in transition time, as shown in Figure 2(b). This is called '*crosstalk slow-down*'.

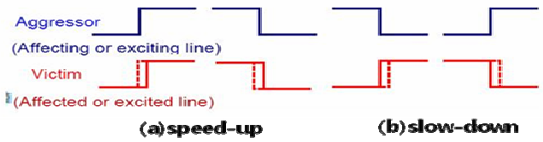

Figure 2. Crosstalk noises (delay)

In addition to glitches and delays, the solution may result in damped voltage oscillations superimposed on top of a glitch or delay, as shown in Figure 3. [4]

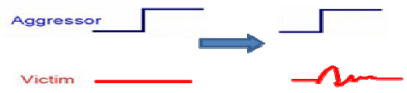

Figure 3.Crosstalk noises (oscillations)

3 A false aggressor pruning algorithm

3.1 A basic idea

The main goal of this paper is to improve crosstalk test by false aggressor pruning. Victim lines are selected on the longest paths. But the number of aggressor lines can be exponential to the number of lines in a circuit. However, many candidates in these aggressors don't need to be considered for crosstalk test. False aggressor detection and false aggressor pruning of the algorithm increase accuracy in noise analysis. The previous approach for noise analysis tool assumes that all aggressor nets may always transition in opposite direction. This leads to increase the complexity and the additional design costs for fixing the noise. Therefore aggressors that have a negligible impact on timing or functionality must to be filtered out of the analysis. The problem that includes the complexity and the cost is resolved by pruning false aggressors. The accurate false coupling interaction between a victim net and an aggressor net is detected and pruned out in fast time for relatively big size circuit.

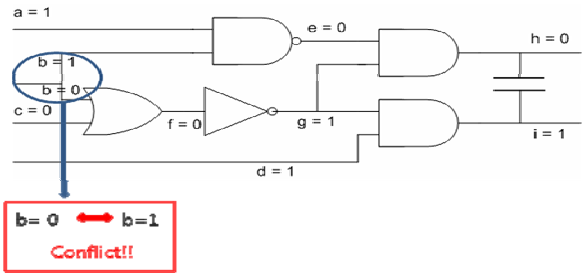

Figure 4. Example circuits(confilct)

Figure 4 shows an example circuit. Assume that 'h' line is an aggressor line and the 'i' line is a victim line. In this circuit, if 'h' has value 0, then the 'i' cannot have value 1. If 'i' has value 1, then 'g, d' must have value 1. Then 'f' must have value 0, and 'b, c' must have value 0. And because 'h' has value 0, 'e' must have value 0. But if 'e' has value 0, b must have value 1. As shown in Figure 4, 'b' cannot have value 1and 0 at same time. It is a conflict. Therefore the circuit cannot have the pattern which makes outputs have (0, 1). As a result, we do not need to consider crosstalk faults concerned about patterns which have the objective (0, 1). Figure 5 shows possible fault lists. We can reduce wrong crosstalk fault candidates because of this reason.

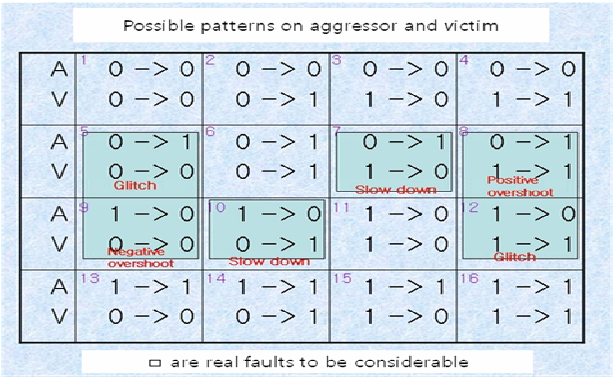

Figure 5. Possible objectives on aggressor and victim

Figure 5 shows possible pairs on aggressors and victims. ‘A’ means aggressor and ‘V’ means victim. Each (0→0), (0→1), (1→0) and (1→1) mean transitions or steady states. Namely, 0 → 1 is a rising transition and 1 → 0 is a falling transition. As shown in Figure 5, six squares are possible fault lists. <5, 12> are glitch, <7, 10> are slow-down, and <8, 9> are overshoot. As an example, as shown in Figure 4, an objective (0, 1) is a conflict. Therefore, <10-slowdown, 12-glitch> can be pruned as shown in Figure 5.

3.2 Previous works

False aggressor pruning algorithm was introduced in [3].

```
Begin
  if ( check_stable_nodevalue (victim, aggressor) )
       Return report_false_coupling_interaction( );
  else {
    Begin
Backward_nodevalue_process(victim, LOGIC_0);
Backward_nodevalue_process (aggressor, LOGIC_1);
       if ( conflict occurred ) pruning_false_aggressor( );
       else {
     Backward_nodevalue_process (victim, LOGIC_1);
    Backward_nodevalue_process (aggressor, LOGIC_0);
       if ( conflict occurred ) pruning_false_aggressor( );
        else report_real_aggressor( );
        }
     End
      }
 End
```

Figure 6. Pseudo code of the previous work

The pseudo code of the false aggressor pruning algorithm is shown in Figure 6. First of all, in the circuit in Figure 4, target aggressor ‘h’ and victim ‘i’ is determined. Since node ‘h’ and ‘i’ has logic 0 or logic 1 by some input vector, backward node value process is activated after logic 1 is assigned on victim node ‘i’ and logic 0 is assigned on aggressor node ‘h’. And the conflict check is performed. If a conflict is presented, then there are not any crosstalk faults when objective (0, 1) is applied. Therefore, the aggressors which are concerned about objective (0, 1) are false aggressors. We can think there is no slow-down fault when objective (0, 1) is applied and the aggressor can be pruned. After that, objective (1, 0) is assigned and the same progress is repeated.

However, there are some problems in this algorithm. The algorithm fails to notice glitches and overshoots crosstalk faults. When objective (0, 1) is assigned, the overshoot-8 and glitch-12 is overlooked and pruned as shown in Figure 5. Similarly, in case the objective (1, 0) is applied, 5-glitch and 9-overshoot is overlooked and pruned. But glitches and overshoots should be taken care about. So, we propose the improved algorithm that is considered about glitches and overshoots for crosstalk test. As a result, the proposed algorithm only prunes the slow-down delay faults.

4 The proposed algorithm

In this paper, the improved algorithm which considers all possible crosstalk faults (slow-down, glitch, overshoot) is proposed. That is, the proposed algorithm is to prune more accurate false aggressors than previous works. The false aggressors never can make slow-down, glitch and overshoot.

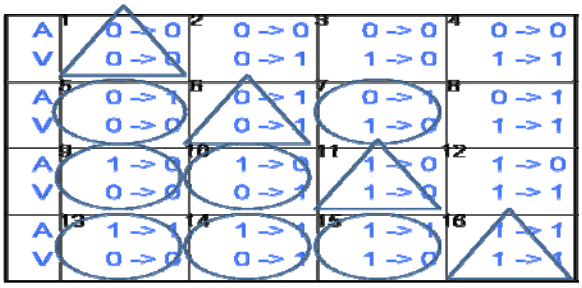

Figure 7. The conflict case in an objective (1, 0)

As shown in Figure 7, if an objective (1, 0) makes a conflict, then the faults in the circles can’t happen. After simulation of the objective (1, 0) is performed, an objective (0, 1) is inserted. If the objective (0, 1) is also a conflict, then <1, 6, 11, 16> triangle values is only possible as shown in Figure 7.

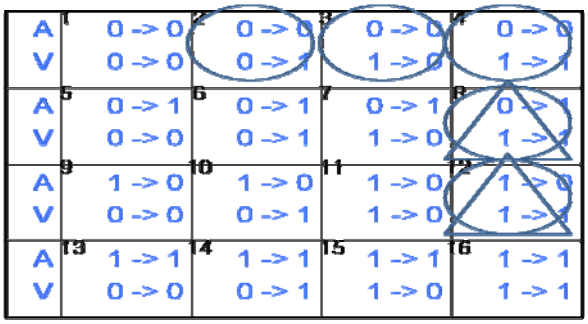

Figure 8. The conflict case in an objective (1, 1) (after (1, 0) and (0, 1) objectives are performed)

And values in circle <2, 3, 4, 8, 12> are not conflict values when objective (0, 1) is inserted but conflict values when objective (1, 0) is applied as shown in Figure 8. At last, about remaining area <2, 3, 4, 8, 12>, an objective (1, 1) is inserted and we check that the conflict is happened or not. And if the conflict is happened, then the aggressors are pruned.

In exactly explanation from start, if the objective (1, 0) is a conflict objective, then <5-glitch, 7-slowdown, 9-overshoot, 10-slowdown, 13, 14, 15> values cannot present. Therefore, we should only take care of 8-glitch and 12-overshoot. And next, the objective (0, 1) is inserted. If a conflict is happened, only <1, 6, 11, 16> are

possible. Therefore, the aggressor is false and pruned. And if the objective (0, 1) is not a conflict but objective (1, 0) is a conflict, then objective (1, 1) which can make 8-glitch or 12-overshoot is inserted. If a conflict is happened, then the aggressor is pruned as shown in Figure 8. If a conflict is not happened, then the aggressor is not pruned. In opposite case, like the above case the progress is performed in order of objective (0, 1) → (1, 0) → (0, 0).

As a result, the proposed algorithm is more accurate than the previous algorithms. Because, the algorithm considers all faults which are slowdowns, glitches and overshoots. The previous algorithms pruned the aggressors which can make crosstalk faults. And pruning accuracy about slowdown of the previous algorithm is also worse than the proposed algorithm.

5 Experiment results

The proposed algorithm was implemented using C language. The experiment was performed on ISCAS-85 benchmark circuits. If a backward search depth is increased, then the number of pruned aggressors is increased. But when the depth is larger than three, the more times are consumed. So, 3 or 5 depth was chosen. In c17 circuit, we tested all possible pairs of aggressor-victim. In other circuits, random pairs of aggressor-victim were tested.

Circuits	**Previous algorithm** (# of prunings/ # of total tests) [3]	**Proposed algorithm** (# of prunings/ # of total tests)
c17	71/90	36/90
c95	78/343	39/343
c432	72/400	46/400
c880	16/2304	8/2304

Table 1. Experiment results (# of prunings/ # of total tests)

Table 1 shows the pruning results. In c17 circuit, we considered the all pair aggressor-victim. Therefore, so many aggressors are pruned. In Table 1, the number of total tests means the aggressor candidates to perform crosstalk test. And the number of prunings means the number of pruned aggressors. Experiments show us the previous algorithm pruned many aggressors which can make crosstalk faults. As an example, in c17 circuit, the difference between 71 and 36 means the number of possible faults about glitch and overshoot. In c880 circuit, the number of pruned aggressors is smaller than other circuits, because of the short depth. If the test depth is increased, then the pruning number is also increased. As a result, we can exactly prune the false aggressor by 10%~30%. So, we can reduce the time consumption of the crosstalk test by 10%~30%. Moreover, if layout information has been used for experiments, the accuracy of the proposed algorithm would be increased because exactly coupling capacitances and net distances could be considered.

6 Conclusions

In this paper, the improved false aggressor pruning algorithm which is based on the backward implication is proposed. Because the proposed algorithm considered the all possible value changes which can make crosstalk faults by coupling capacitance, more accurate pruning results could be achieved compare to the previous one. Throughout the proposed algorithm, we could obtain accurate results quickly for relatively huge size circuits. As a result, average 10%~20% false aggressors could be pruned in more than 10% reduced time consumption that is compared to the previous algorithm.

Acknowledgment

The authors would like to appreciate CAD tool support from IC Design Education Center (IDEC).

References

[1] A. B. Kahng, S. Muddu and D. Vidhani, “Noise and Delay Uncertainty Studies for Coupled RC Interconnects”, Proceeding of IEEE International ASIC/SOC Conference, pp. 3-8, 1999.

[2] Tong Xiao, Chih-Wei Chang, Marek-Sadowska, M., “Efficient static timing analysis in presence of crosstalk”, Proceeding of IEEE International ASIC/SOC Conference, 13-16, pp.335-339, 2000.

[3] Hyungwoo Lee and Juho Kim, “False Aggressors pruning using Path sensitization and logic implications ”, Proceeding of Asia-Pacific Conference on ASIC, pp.278-281, 2004.

[4] Jae-Seok Yang, Jeong-yeol Kim, Joon-Ho Choi, Moon-Hyun Yoo and Jeong-Taek Kong, “Elimination of false aggressors using the functional relationship for full-chip crosstalk analysis”, Proceeding of the Fourth International Symposium on Quality Electronic Design, pp.344-347, 2003.

[5] Hiroshi Takahashi, Keith J. Keller, Kim T. Le, Kewal K. Saluja and Yuzo Takamatsu, “A Method for Reducing the Target Fault List of Crosstalk Faults in Synchronous Sequential Circuits”, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, pp.252-263, 2006.

[6] Sandip Kundu, SujitT. Zachariah, Yi-Shing Chang, and Chandra Tirumurti, “On Modeling Crosstalk faults”, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, pp.1909-1915, 2005.