XSIM: An Efficient Crosstalk Simulator for Analysis and Modelling of Signal Integrity Faults in both Defective and Defect-free Interconnects

Ajoy K. Palit, Kishore K. Duganapalli, Walter Anheier
University of Bremen, ITEM, Otto-Hahn-Allee-1, D-28359 Bremen, Germany.
E-mails: {palit|kishore|anheier}@item.uni-bremen.de

Abstract—The paper presents an efficient crosstalk simulator tool “XSIM” and its methodology developed by the signal integrity fault modeling and test research group of ITEM, University of Bremen, for analysis and modeling of signal integrity faults in deep sub-micron (DSM) chip. The tool can be used for analyzing the crosstalk coupling behavior in both defective and defect-free parallel interconnects. Using the XSIM tool one can also determine the critical values of various interconnect’s parasitics and the critical values of crosstalk coupling capacitance and resistive bridging (i.e. mutual conductance, if any) beyond which device will most likely suffer from the signal integrity losses, whereas for lower coupling values device will continue to behave as crosstalk fault tolerant. The special features of this simulation tool are that it is based on an indigenous methodology implemented in C++ and Microsoft Foundation Classes (MFC) and can be run on any standard PC with Windows 2000 or Windows XP operating system. The advantage of such implementation is that it provides a user-friendly Graphical-User-Interface (GUI) which not only makes the tool very easy-to-use but also flexible and at least 11 times faster than commercial circuit simulator like PSPICE and provides very accurate simulation results which are very close to the one obtained from latter.

Key words: XSIM, Crosstalk effect, Signal Integrity faults, Aggressor-victim, Resistive Bridge.

1. INTRODUCTION

The need for an efficient i.e. flexible, easy-to-use, fast and yet very accurate tool for signal integrity fault analysis is well understood by the current SoC designers in this deep sub-micron era. This is because in DSM circuit designs, the interconnect parasitics and also the coupling capacitance between the adjacent interconnects have become significant as the wires become taller and narrower while the distance between them decreases. Due to these changes, signal integrity faults due to crosstalk noise between the physically adjacent nets and also due to interconnect RLGC parasitics have become an important concern [1]. Signal integrity fault caused due to the distributed interconnect parasitics and coupling noise between the two adjacent wires can lead to various functionality problems such as, crosstalk delay, logic hazards and even early arrival of signal on the affected net of the device. The affected net is known as the victim and the neighboring switching net is usually called aggressor.

As per [2] accurate computation of the delay change in the victim net due to crosstalk is very important in the circuit design process. With the above objective in mind Li, et al has developed the XtalkDelay [2], which is a crosstalk aware timing analysis tool for chip-level designs. Such tool employs a path-based approach; uses detailed and accurate distributed RC parasitics for critical nets and their aggressors; uses BSIM3-accurate gate models and invokes finally HSPICE for delay computation using only the minimum required set of input patterns. While with similar objective in mind, in addition to crosstalk glitch height computation, in this paper we propose alternatively an efficient indigenous tool “XSIM” and its methodology using the distributed RLGC parasitics, distributed coupling capacitances, mutual conductances etc., obtained by parameter extraction tool, of the concerned interconnects pair.

Since XSIM is developed based on indigenous methodology and more importantly it does not involve any commercial software like HSPICE or PSPICE it is needless to mention that XSIM is much faster (at least 11 times) and yet accurate enough like PSPICE simulations. Furthermore, “XSIM” takes into account the length of interconnects and also the RC linear models of the CMOS drivers and receivers of victim and aggressor and all possible input signal combinations. The tool is flexible because using the same graphical window of the XSIM one can select any possible combination of two input signals including the skewed ones in order to simulate and compute the various crosstalk coupling effects between the aggressor-victim interconnects. The tool is, furthermore, new in the sense that the simulator is developed based on the distributed ABCD crosstalk model of aggressor-victim interconnects which considers both coupling capacitance and (very high) resistive bridging between the aggressor-victim interconnects, besides consideration of usual RLGC parasitics based distributed interconnect model as discussed in Section 2. Section 2 also describes the brief methodology or algorithm that made the tool very fast and flexible followed by simulation experiments carried out with XSIM tool and discussion in Section 3. Finally the paper concludes with brief remarks presented in Section 4.

2. METHODOLOGY AND TOOL’S DESCRIPTION

The XSIM tool is developed based on the distributed crosstalk ABCD models of the pair of interconnects [4]. All interconnects’ parasitics including the coupling capacitance and resistive bridging parameter values etc. are extracted through Philips CAT-Bridge extraction tool from 130nm technology device. The advantage of using such distributed models is that because of the consideration of interconnects’ distributed RLGC parasitics i.e. resistance, self inductance, conductance and capacitance etc. and also distributed coupling capacitance and distributed mutual conductance between the concerned aggressor and victim nets, the generated model is very accurate. Further advantage of using such model is that
cascade of identical \( n \) number of ABCD models is the \( n \)th power of individual ABCD matrix.

In order to illustrate the methodology of XSIM tool Fig. 1 and Fig. 3 respectively show the distributed models of two defect-free and defective parallel interconnects. In Fig. 1 each dotted rectangular block of length \( \Delta x \) containing interconnects’ RLGC parasitics and also the mutual capacitance and mutual conductance can be represented by distributed ABCD model as shown in Fig. 4. Fig. 2 below shows a pair of defective interconnects due to bad etching. In order to characterize the behavior of the defect both the coupling capacitance and shunt conductance are assumed to be several times bigger in the defective parts than the usual value in the defect-free case. Also note that in the defective part of interconnects where there is a complete resistive contact between the two interconnects, the shunt conductance in that particular point will be much greater than other points in the defective part because of the resistive bridging, whereas coupling capacitance for that particular point will be close to zero due to resistive short. Latter case is similar to a parallel plate capacitor with a resistive short between the two plates, i.e. a highly leaky capacitance.

Also, it is to be emphasized that here under-etching defect is assumed to have affected only \( p \) blocks (each of length \( \Delta x \)) of right-L distributed RLGC models from both top and bottom interconnects and for these \( p \) blocks the coupling capacitance values are \( k_i \) times of usual coupling capacitance value \( (c_{n\Delta x}) \) and similarly, the shunt conductance values for these defective \( p \) blocks are \( w_i \) times of usual shunt conductance value \( (g_{n\Delta x}) \), where \( i = 1, 2, 3, \ldots, p \). Furthermore, Fig. 3 depicts the distributed RLGC model for the full length of defective pair of interconnects.

Note that in Fig. 3 the effects of under-etching defect, i.e. increased coupling capacitance and/or increased shunt conductance in defective \( p \) blocks, have been shown to be distributed all throughout the interconnects length, which resulted in \( k c_{n\Delta x} \) and \( w g_{n\Delta x} \) values respectively for all coupling capacitances and shunt conductances in all \( n \) blocks, where the respective value of \( k \) and \( w \) is

\[
k = \left( \frac{n - p}{n} + \sum_{i=1}^{p} \frac{k_i}{n} \right), \quad \text{and} \quad w = \left( \frac{n - p}{n} + \sum_{i=1}^{p} \frac{w_i}{n} \right).
\]

Also note that because of this kind of distribution of defect’s effect our model will generate always same output irrespective of defect’s location on the interconnects. In fact, such distribution of defect’s effect holds when the defect is located exactly on the middle of both interconnects. In addition to the above, the linear RC models of CMOS drivers and capacitive loads of CMOS receivers of both aggressor and victim nets are considered in Fig. 4.
The fourth order approximation of \( T_j(s) \) term from the numerators is:

\[
T_j(s) = t_{j0} + t_{j1}s + t_{j2}s^2 + t_{j3}s^3 + t_{j4}s^4; \quad j = 1, 2, 3, 4.
\]

Obviously, terms \( t_{j0} \) through \( t_{j4} \) are functions of interconnect parasitics including the terms mutual capacitance and mutual shunt conductance and linear parameters of drivers and receivers’ models. Each term from \( t_{j0} \) through \( t_{j4} \) is obtained by summing one or more convergent infinite series.

Fig. 5 describes the flow chart of the algorithm of our XSIM tool. Initially the algorithm calculates the coefficients (terms \( t_{j0} \) through \( t_{j4} \)) with the input parameters of interconnects, drivers and receivers. Now depending on the user input signal combination, the algorithm substitutes the appropriate signals in s-domain for aggressor and victim inputs in (1) and (2). Thereafter by finding the partial coefficients, the algorithm applies partial fraction method on (1) and (2), followed by inverse Laplace transformation to find time domain representation of output signal waveforms of victim and aggressor respectively (See [3] for detailed procedure). The time domain representations of both waveforms are very important as they provide much useful information about the transmission signal quality (integrity) over long interconnects. Finally the algorithm simulates and gives the output waveforms within the user required time span.

Fig. 6: Graphical-User-Interface of XSIM tool

The entire algorithm is implemented in C++, whereas the user-friendly Graphic interface (GUI) is developed using the MFC as shown in Fig. 6. Under the “Edit” menu, the tool provides some default values of the parameters related to interconnects, drivers, receivers of a given technology. The tool also provides choices of input signals and simulation time etc. However, the above parameter values can be changed as per the requirement of user. In addition to the above, non-zero time delay for the input signal represents the skewed input signal, whereas time constant represents the steepness of the rising or falling input signal. However, for static “0” and static “1” signals on the victim input the time delay and time constant have no effect. It is to be noted that because of non-disclosure agreement with Philips Semiconductors, Design Technology Center (DTC), Hamburg, actual defaults values of interconnect parameters used by our “XSIM” tool are deliberately changed and the displayed values are not at par with the Philips CMOS12 (130nm) Technology.

![Fig. 7: Illustration of 'Run & Plot' button of XSIM.](image)
Once the interconnect length and per-unit-length interconnect’s parameter values of resistance \( r \), self inductance \( l \), conductance \( g \), capacitance \( c \) and also the mutual capacitance \( c_{m} \) and mutual conductance \( g_{m} \) are all selected besides the selection of driver and receiver parameters and also the input of aggressor and victim (radio buttons) and the simulation time etc. the “**run & plot**” button can be clicked to observe the result graphically (see Fig. 7). The corresponding graphical plot eventually shows input and output signals of both aggressor and victim nets along with victim net’s delay or crosstalk glitch height produced etc. if any on the victim net. Furthermore, clicking the “**Browse**” button one can also select any text file such as “PSPICE_Result.txt” file with it’s appropriate path for plotting results too. This option is quite helpful in plotting the simulation result obtained from any commercial simulator like PSPICE, for comparison purposes, for instance. Fig. 7 shows the purpose of “Run & Plot” button in our XSIM tool. Fig. 7 also shows the crosstalk simulation results with our “XSIM” tool for rising input signal on aggressor and victim with static zero for some arbitrary settings of “XSIM” parameter values.

### 3. SIMULATION EXPERIMENTS WITH XSIM

In this section we describe a few simulation results carried out using our XSIM tool for analysis of signal integrity faults. All simulations are usually carried out using the Philips CMOS12 (130nm) technology parameters for the metal layer 2. However, occasionally we have varied certain parameter of interconnects or driver/receiver in order to study the influence of that parameter on the signal integrity faults. For all simulation experiments we have selected the interconnect length as 200 micrometer, however, if required it can be set to any other value.

Fig. 8 shows the crosstalk simulation experiments of aggressor and victim nets where aggressor input is driven by a rising signal and victim output is driven by falling (opposite) transition. These options are selected by the radio buttons provided under the aggressor and victim group box in XSIM tool. Time constants of both input signals are also selected as 10 ps, implying that signal reaches the 63.2% of desired \( V_{dd} \) (or \( V_{ss} \)) in 10 ps. Therefore, smaller time constant implies the faster transition. Delay times in both inputs are selected as zero, implying that we have non-skewed input signal for both nets. Furthermore, driver and receiver strengths are selected to be same for both nets.

Simulation results depicted in Fig. 8 shows that because of identical size of drivers and receivers of both nets outputs are equally delayed. This is clear from 50% delay times of both output signals which are approximately 20 ps in this case. Similar experiment was also carried out as depicted in Fig. 9 using the same choices of input signals and other parameters except for much higher values of mutual capacitance and mutual conductance. Fig. 9 depicts that both victim’s delay and aggressor’s delay in this case have very much increased because of 10 times larger mutual capacitance and also 100 times larger mutual conductance than that in Fig. 8. However, both delays are same because of identical choices of drivers and receivers for both nets. The 50% delay for both nets in this case is greater than 40ps.

![Fig. 8: Delayed output signals of aggressor and victim nets.](image)

![Fig. 9: Larger delayed output signals of aggressor and victim nets.](image)

Further simulations are also carried out and depicted in Fig. 10 with falling input transition on victim and rising input transition on aggressor, however, in this case aggressor input is a skewed one with time delay approximately equal to 5ps. Also, simulation experiments depicted in Fig. 10 considers the different coupling capacitances ranging from their nominal
value until 20 times of nominal value. For convenience, in this case, aggressor’s output is not shown. Fig. 10 shows three different victim’s outputs corresponding to coupling capacitance value \( c_m \), \( 10c_m \), and \( 20c_m \). Note that because of skewed input signal of the aggressor, victim’s output is very much distorted in addition to usual delay, and the amount of distortion is more when the influencing factor (i.e., coupling capacitance value) is larger. This simulation result is clearly very different from the one in Fig. 9, as the latter depicts a smooth victim’s output.

Several other experiments are also carried out with rising input transition on the aggressor but victim with static “0” signal. For instance, Fig. 11 below depicts the crosstalk simulation using our XSIM tool when the aggressor has rising input transition and victim has static zero signal.

In this case also coupling capacitance and mutual conductance values selected for simulation are approximately 10 times larger than the usual coupling capacitance value and 100 times larger than the mutual conductance value respectively. Notice that here because of large values of coupling capacitance and mutual conductance more crosstalk glitch is produced on the victim’s output signal. The generated crosstalk glitch height is significantly large (approximately 280mV) enough to produce signal integrity faults. This is because the glitch height (and its duration is large too) is likely to exceed the permissible upper limit of logic low threshold for CMOS 12 technology. Similar to the case in Fig. 9, here, also large value of mutual conductance does not allow the crosstalk glitch to finally settle down to it’s steady state value \( V_{ss} \) and thereby, rendering a constant steady state error in the victim’s output signal.

On the other hand, simulation results shown in Fig. 12 illustrate the effect of increasing coupling capacitance on the crosstalk glitch height. It is further shown that when the coupling capacitance is increased by 10 times and 20 times of it’s nominal value, besides the increase of crosstalk glitch height the time of occurrence of maximum glitch height is also shifted more towards right.

In this case also coupling capacitance and mutual conductance values selected for simulation are approximately 10 times larger than the usual coupling capacitance value and 100 times larger than the mutual conductance value respectively. Notice that here because of large values of
Similar to Fig. 12, simulation results shown in Fig. 13 illustrates the influence of increasing mutual conductance (decreasing bridging resistance) on the steady state value. Note that here the aggressor input is driven by rising transition whereas victim is driven by static “0” signal and 10 times of nominal coupling capacitance is considered. Only the victim outputs for three different values of mutual conductance are shown in Fig. 13 for the visibility of the effect. Here the crosstalk glitch height on the victim is not much influenced instead the final steady state value of the signal on victim is much influenced. So the increase of mutual conductance or decrease of the bridging resistance leads to signal integrity faults in capturing time of the signal.

Finally, the last simulation experiment depicted in Fig. 14 shows the comparison of crosstalk simulation performed with the commercial software like PSPICE. It can be seen from the Fig. 14 that in this case also aggressor input is driven by rising transition and victim input is a static zero signal as in the previous case. Also, all other simulation parameters applied to PSPICE simulation and XSIM tool are identical.

From the Fig. 14 it is very clear that the victim’s output signal i.e., the simulated crosstalk glitch obtained from the XSIM tool closely overlaps with the results obtained from the PSPICE, implying that the accuracy of XSIM simulator is highly comparable with the PSPICE. However, our XSIM tool performs the same simulation at much higher speed, which is at least 11 times faster than that of PSPICE. Apart from the above a few other simulations can be carried out to illustrate that crosstalk glitch height is also very much influenced by the aggressor’s driver strength or time constant of the input signal. The effect of interconnect length on the delay of the signal is also simulated using the tool. Likewise, influence of self-inductance on the signal oscillation or over-shoot/undershoot can also be demonstrated using the XSIM tool, which in turn can help in determining the critical values of the same.

In this paper a new crosstalk simulator tool namely, “XSIM” is developed which can be efficiently utilized for analysis of various signal integrity faults, such as 50% delay timing violation, logic hazards etc., caused primarily due to crosstalk coupling capacitance and mutual conductance (if any) in both defective and defect-free parallel interconnects. The XSIM tool is based on the distributed ABCD models of parallel interconnects, in which linear RC models of drivers and receivers are also taken into account besides interconnects’ per-unit-length RLGC parasitics and mutual capacitance and mutual conductance. The tool is implemented in visual C++.

Various crosstalk simulations were carried out and illustrated using graphical plots. It has been observed that effect of large coupling capacitance is two-fold, as it produces timing delay violation when aggressor and victim nets are driven by mutually opposite transitions, whereas it produces crosstalk glitch when the victim has either static “0” or static “1” signal while aggressor has either fast rising or falling transition. On the other hand, the influence of mutual conductance is mainly on the final steady state value of the victim’s output signal, as it does not allow the output signal to finally settle down to it’s original steady state value, rendering it to have a permanent steady-state error. Conducting the crosstalk simulation experiments using the tool one can easily determine the critical values of various influencing parameters below which the device will likely to behave as crosstalk fault tolerant one. The C++ and MFC based implementation of the tool also provides a user-friendly GUI which makes the tool not only very easy-to-use but also flexible, fast and yet very accurate enough like the PSPICE commercial simulator. Like all other scientific works this tool also suffers from certain short falls as the tool currently cannot accept the mutual inductance value between the pair of nets, which is currently left as the future perspective of this work. Apart from the above, the crosstalk simulation of multi aggressor-single victim feature needs to be also incorporated into our future version of XSIM tool, although the same feature is readily available in MATLAB source code.

5. REFERENCES

4. CONCLUSIONS AND FUTURE WORK

Fig. 14: Comparison of XSIM and PSPICE simulations results when aggressor input is a rising transition and victim has static zero input.