Nostra-XTalk : A Predictive Framework for Accurate Static Timing Analysis in UDSM VLSI Circuits *

Debasish Das, Ahmed Shebaita, Yehea Ismail, Hai Zhou and Kip Killpack†
Department of Electrical Engineering and Computer Science, Northwestern University, Evanston, IL 60208
† Strategic CAD Lab, Intel Corporation, Hillsboro, OR 97124

ABSTRACT

This paper presents a predictive framework for accurate static timing analysis in UDSM VLSI circuits. As technology scales to smaller dimensions, coupling capacitances are becoming a critical factor in signal integrity analysis. Coupling capacitances contribute to the noise and play a seminal role in determining the timing windows of a circuit. Accurate analysis of coupling effects is indispensable for meaningful static timing and signal integrity analysis. Our proposed framework presents a Directed Search technique to calculate accurate coupling effects. We performed experiments on the ISCAS'85 benchmarks and present the accuracy improvement up-to 45.5% compared to existing approaches. We also show that our framework decreased cell delay look-up table accesses up-to 64.8%. Our results present the coupling effect on static timing analysis.

Categories and Subject Descriptors: J.6 [Computer-Aided Engineering]: Computer-Aided Design
General Terms: Algorithms, Performance
Keywords: Static timing analysis, Crosstalk, Modeling

1. INTRODUCTION

Progress of deep sub-micron technologies has resulted into shrinking geometries which in-turn, have led to a decrease in the self-capacitance of wires. Meanwhile it also increased the aspect ratio of wires over 2.0 for intermediate wiring layers [13]. Future trends indicate that the lateral component of interconnect capacitance (coupling capacitance) can reach five times [13, 9] as much as the vertical capacitances. Coupling capacitance increases the total equivalent capacitance of interconnects. Mathematically

$$C_{eq} = C_g + \sum_{\forall j \in coupledlines} MCF_j C_{cj} \quad (1)$$

where, C_g is the equivalent line-to-ground capacitance, C_{cj} is the coupling capacitance between the line of interest and its j_{th} neighbor, and C_{eq} is the equivalent total capacitance of the line of interest. MCF_j is the miller coupling factor that depends upon the relative switching activity of the line of interest and its j_{th} coupled line. Recent researches have calculated the switching factors to be 0 to 2 when the waveform on coupled interconnects are assumed to be pulses [12], -1 to 3 when the waveforms are assumed to be ramps [8, 2],and -1.885 to 3.885 when the waveforms are considered exponential [7].

Coupling capacitances contribute significantly in calculating interconnect delays. Due to the dominance of coupling capacitance over all parasitics of interconnect wire, failure to take the accurate effect of coupling (henceforth will be called *crosstalk* interchangeably) into timing or signal integrity analysis will produce results far off from the reality. Also assuming a simple coupling model with 0 to 2 switching factor is not accurate. Widespread use of input slews in characterizing logic gates (standard ASIC design flow) motivates using a ramp based model for accurate switching factor computation.

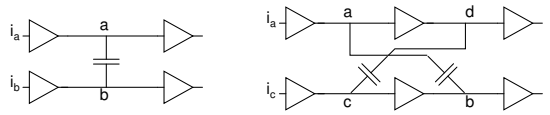

Figure 1: Timing analysis with crosstalk is a mutual dependence problem: (a) local problem; (b) global problem

Timing and crosstalk effects are mutually dependent [15]. For example, consider the two coupled nets in Figure 1(a). The switching time on net a is dependent on the switching time on net b. But the switching time on net b is not fixed, it is dependent on the switching time on net a. Subsequently all state of the art approaches to static timing analysis crosstalk are iterative. Starting with input arrival time and an initial crosstalk effect, the timing information of the interconnects are iteratively updated until converged to a stable situation.

Chaotic iterative scheme [15] gives a theoretical insight into any iterative procedure used for static timing analysis. Final timing windows are fix-points of a lattice as presented in [15]. We present a new timing algorithm that makes use of the directed search technique presented to predict the worst and best coupling effects from a given input arrival time and an initial crosstalk effect and thus decouples the whole circuit for timing analysis. We also establish that the directed search does an inherent fix-point iteration to come up with accurate coupling effect.

We divide our switching windows into time slots due to reasons presented in [3]. In comparison to [3] our approach is more accurate due to application of accurate coupling model on nonlinear gate delay models based on standard cells from [4]. Our framework is amenable to state-of-the art static timers and can be integrated easily into any existing signal integrity analysis flow. In essence *N*ostra-XTalk has the following novel components.

1. Directed search technique for coupling effect analysis
2. Accurate static timing analysis with crosstalk

The rest of the paper is organized as follows. Section 2 presents motivation for efficient and accurate coupling anal-

*This work is supported by a grant from Intel Corporation

GLSVLSI'07, March 11–13, 2007, Stresa-Lago Maggiore, Italy.

ysis. Section 3 presents the directed search technique. Section 4 presents the convergence conditions in the fixed point iteration of directed search. Section 5 presents the accurate timer algorithm using directed search. We present our experimental results in Section 6. Conclusions and future work are described in Section 7.

2. MOTIVATIONAL EXAMPLE

We now use an example shown in Figure 2 to motivate the need for accurate crosstalk analysis during timing analysis . Nets $I1$ and $I2$ fan into input pin of gates $G1$ and $G2$. Rise and fall delay windows on net $I1$ are [2,4] and [2.5,3.5]. Similarly on net $I2$ the windows are [3,5] and [3.5,4.5]. $G1$ and $G2$ are inverters of different sizes which leads to different timing arcs arc1 and arc2 in them. Each arc contains four look-up tables which corresponds to rise delay, rise transition, fall delay and fall transition respectively. Also assume that net $I1$ and $I2$ has similar rise and fall slew windows given by [0.2,0.6] and [0.4,0.8]. Such delay and slew windows are obtained from a timing flow as presented in [5]. Note the difference of our notations from prevalent conditions. Rise and fall delay windows give 0% arrival time rather than 50% point. Rise and fall slew windows represents the time taken for the signal to transition between 0% to 100% of V_{dd}. In Figure 2 we have not shown the ground capacitance. Let's call the ground capacitance as C_g^i for each gate i. C_g^i is the lumped capacitance corresponding to net O^i. For sake

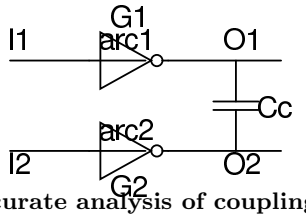

Figure 2: Accurate analysis of coupling capacitance

of simplicity assume we are doing a worst MCF computation for rise delay window on net $O1$. Without any loss of generality consider net $O1$ as victim and $O2$ as aggressor. For the worst MCF computation, we need to consider the rise delay window at $I1$, rise slew window at $I1$, fall delay window at $I2$ and fall slew window at $I2$ respectively. We define a timing event as an ordered pair of arrival time and slew.

Consider a coupling model based on slew and arrival times recently proposed by [5]. Assuming delay on arc1 due to input rise slew as 0.6 and loading cap of C_g^1+C_c be 0.8. Input window [2,4] transforms to [2.8,4.8]. Let the fall delay on arc2 due to input fall slew as 0.4 and loading cap of C_g^1+C_c be 0.6. Input window [3.5,4.5] transforms to [4.1,5.1]. Slew selection are based on the conventions of static timing analysis [14]. Similarly due to the increase in loading cap, the slew windows at $O1$ and $O2$ also change. Assume rise slew at $O1$ changes to 0.7 from 0.6. Similarly fall slew at $O2$ changes to 0.5 from 0.4. Following the coupling model from [5], we get the worst MCF from timing events $V = (4.8, 0.7)$ and $A = (4.8, 0.5)$ where V and A denote victim and aggressor respectively. Worst MCF is computed by $1 + 2k$ where $k = \frac{0.5 t_v^s}{t_a^s}$(using formulation from [5]). We have $t_v^s = 0.7$ while $t_a^s = 0.5$. Thus we get MCF as 2.4. Using this we update the total capacitance as $C_g^1 + 2.4 \times C_c$. We compute the delay due input rise slew as 0.6 and loading cap of C_g^1 + $2.4 \times C_c$ and update the input window [2,4] to [3.0,5.0].

The inherent assumption in the analysis presented above is ignoring the correlation between input timing event and worst MCF. Even if we consider slew selection based on static timing analysis convention, each timing event from input window will result in different MCF. Suppose there are two timing events $T1 = (3.9, 0.6)$ and $T2 = (4.0, 0.6)$ from input window [2,4]. $T1$ resulted in MCF of 2.5 while $T2$ gave MCF of 2.4. But delay push-out due to $T1$ can still be less than the delay push-out due to $T2$ and thus the event $T2$ still provides the worst MCF of 2.4. If it is not then the analysis presented above can give wrong result. It is important to verify the delay push-outs due to each timing event from input window before we choose the worst MCF.

Based on the discussion above, for worst MCF computation, searching the space spanned by input delay and slew windows are necessary. Our directed search approach is based on this intuitive idea and we present an efficient approach to search for the input timing event which results in the worst delay push-out rather than trying to find a worst coupling factor.

3. DIRECTED SEARCH APPROACH

It was also shown in [8] that the exact MCF is a function of the overlap scenario between the victim and aggressor signals. Four of the possible overlap scenarios are shown in Figure 3. Figure 4 shows the relations that govern the MCF

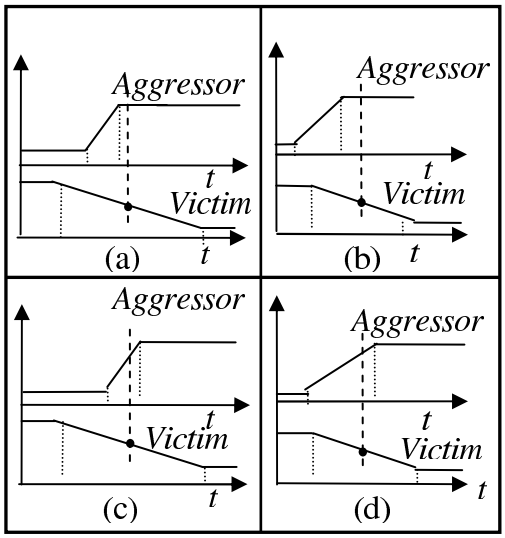

Figure 3: Possible overlap scenarios

for both the victim and aggressor in each of the four cases shown in Figure 3. Figure 4 shows the MCF of both victim and aggressor for opposite and similar switching cases such that k is the ratio of aggressor waveform that overlap with that of the victim up till the victim's threshold voltage v_{th}, t_a^s is the aggressor output slew and t_v^s is the victim output slew. The '+' operator corresponds to the opposite switching case while the '-' operator corresponds to the similar switching case. k is similar to overlap ratio as defined by [5]. Strict lower and upper bounds of 0 and 1 holds for overlap ratio. Bounds of k plays an important role in our technique as mentioned in Section 3.1.

Two windows are to be determined at the output of each gate. The first window is the arrival delay window, $[D_0^l, D_0^h]$, defined as the possible output delay values after which the gate output starts to change from its steady state. The second window is the slew window, $[s_0^l, s_0^h]$, defined as the

	Victim MCF	Aggressor MCF
a	1±2k	$1\pm\frac{t_a^s}{t_v^s}$
b	1±2k	$1\pm\frac{t_a^s}{t_v^s}(2k-1)$
c	1±2k	$1\pm\frac{t_a^s}{t_v^s}$
d	1±2k	$1\pm\frac{0.5}{k}$

Figure 4: Victim and aggressor MCF for various coupling scenarios

window of the possible slews at the output of the gate. These two windows are computed based on the given delay and slew windows at the input of both the victim and aggressor. Figure 5 shows a circuit diagram for a victim, aggressor, the coupling capacitance between them, and the slew and delay intervals.

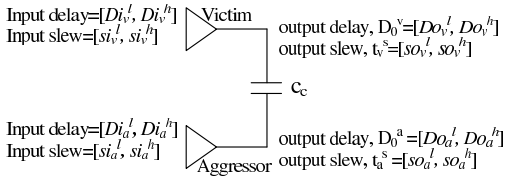

Figure 5: Victim and aggressor equivalent circuit

Figure 6 shows the equivalent decoupled circuit of that shown in Figure 5. The output delay at the victim and aggressor gates is given by

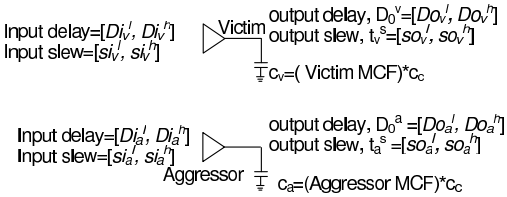

Figure 6: Victim and aggressor decoupled circuit

$$D_o^v = D_i^v + t_v^d \tag{2}$$

$$D_o^a = D_i^a + t_a^d \tag{3}$$

where, $D_i^v \in [Di_v^l, Di_v^h]$, t_v^d is the delay through the victim gate, $D_i^a \in [Di_a^l, Di_a^h]$ and t_a^d is the delay through the aggressor gate.

A cell library is used that has slew/delay tables associated with each gate for both rising and falling output waveform. These tables are looked up with the input slew and the output effective capacitance and give both the corresponding output delay and the output slew. It was observed that the output delay and slew of this cell library is linear with the input slew and load capacitance. This observation can be explained by assuming the α power law delay and slew approximations.

The output delay, t^d, and the output slew, t^s, as functions of the output capacitance and input slew are given by [11]

$$t^d = (\frac{1}{2} - \frac{1-\frac{v_{th}}{V_{DD}}}{1+\alpha})t^s + \frac{C_L V_{DD}}{2I_o} \text{ and} \tag{4}$$

$$t^s = \frac{C_L V_{DD}}{I_o}(\frac{9}{8} + \frac{V_{D_o}}{0.8V_{DD}}\ln(\frac{10V_{D_o}}{eV_{DD}})) \tag{5}$$

where C_L is the output capacitance, α is the velocity saturation index, V_{D_o} is the drain saturation voltage at V_{GS}=V_{DD}, I_o is the drain saturation current at V_{GS}=V_{DD}. It is clear from Equation 4-5) that the output delay and the output slew are linearly related to the output capacitance and the input slew. Thus the library tables for both the victim and aggressor gates can be visualized as shown in Figure 7.

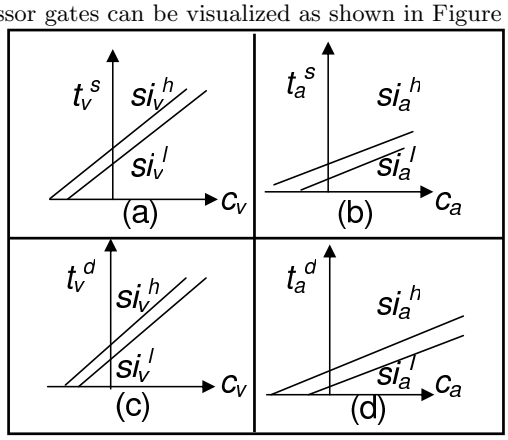

Figure 7: Plot of aggressor and victim output delays and slews

Figure 7 shows the linear relation between the output delay and slew and the output capacitance for a given input slew range $[s_i^l, s_i^h]$. To determine worst/best delay and slew each of the four possible cases shown in Figure 3 should be considered. The worst/best case delay computed based on each case should be compared with those computed from the other cases to find the ultimate worst/best delay and slew. The following section describes the details of obtaining the worst case delay assuming the scenario shown in Figure 3(a).

3.1 Worst case delay computation

The worst case computation involves spanning the possible values of k that ranges from 0 to 1. For each value of k the victim capacitance is computed as $(1+2k)C_c$ as shown in Figure 4. This value of the victim capacitance results in a set of values for the victim delay and slew as shown in Figure 8.As shown in Figure 4, the aggressor capacitance for this case(a) is

$$c_a = (1 + \frac{t_a^s}{t_v^s})c_c \tag{6}$$

Thus, the equation that gives the aggressor slew in terms of the aggressor effective capacitance c_a is given by Equation 7

$$t_a^s = t_v^s(\frac{c_a - c_c}{c_c}) \tag{7}$$

where t_v^s is obtained as shown in Figure 8(a). Figure 9(a) shows the plot of Figure 8 on top of Figure 7(b).

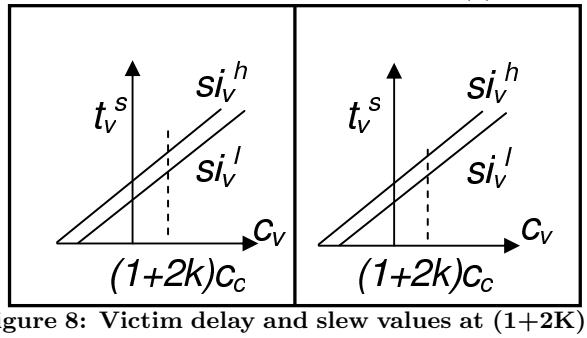

Figure 8: Victim delay and slew values at (1+2K)C_c

The intersection point between Figure 8 and Figure 7(b) give the aggressor capacitors $[c_1, c_2]$. Each aggressor delay points, t_a^d, can be obtained by substituting with one

of the aggressor capacitors [c_1,c_2] as shown in Figure 9(b). Each set of the values obtained, t_v^s, t_v^d, t_a^s, t_a^d, should be tested to see whether it satisfies the overlap scenario shown in Figure 3(a). The sets that satisfy the scenario are then compared and the worst delay and slew that can occur in this scenario is determined. The previous computation steps should be repeated with all the possible scenarios shown in Figure 3(a)-(d) and the worst out of all the scenarios is the absolute worst case delay.

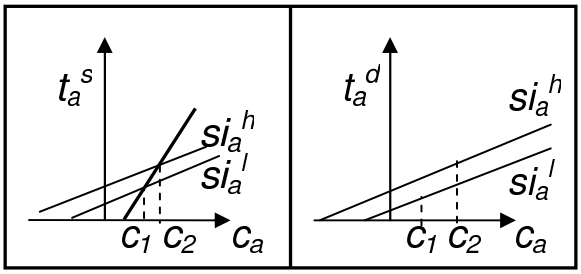

Figure 9: Graphical solution for aggressor slew and delay

4. CONVERGENCE

Directed search for 2 coupled nets is presented in Section 3. But in a general circuit a net is coupled with more than one nets. As shown in Figure 10, net O1 is coupled with net O2 with coupling capacitance C_c and net O3 with coupling capacitance C13, net O2 is coupled with O1 and O3 while net O3 is coupled with O1 and O2. Application of directed search on a cluster of more than one coupled nodes is of practical interest.

4.1 Circuit modeling

Signal propagations in a circuit forms a directed acyclic graph (DAG) while crosstalk couplings form bidirectional edges in the DAG. Formally we model our circuit as a general directed graph $G = (V,E)$ where G is partitioned into two subgraphs $G^C = (V, C)$ and $G^F = (V, F)$. The edges in G^C are the bidirectional coupling edges and the edges in G^F are the fan-in edges. The gates of the combinational circuit are represented by the elements of set V. Graph G represents a timing graph with superimposed coupling graph. Each gate has rise delay window (rd_i^l, rd_i^h), fall delay window (fd_i^l, fd_i^h), rise slew window (rs_i^l, rs_i^h) and fall slew window (fs_i^l, fs_i^h) respectively.

DEFINITION 1. *k-cluster is defined as the k-tuple* $< V, A^0, ..., A^{k-2} >$ *where*

$$V, A^0, A^1, ..., A^{k-2} \in V \tag{8}$$

$$\forall i \in (0, ..., k-2) : (V, A^i) \in C \tag{9}$$

V is called as the victim node while A^i are its aggressors.

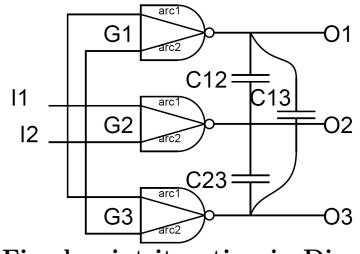

Figure 10: Fixed point iteration in Directed Search

Based on Definition 1 we formulate our problem

PROBLEM 1 (DIRECTED SEARCH ON K-CLUSTER). *Given a k-cluster with initial 1C switching windows, find the best and worst case coupling capacitance that appears on victim V due to aggressors $A^i : i \in (0, ..., k-2)$.*

4.2 Best and worst case coupling capacitance as fix-points

Given a k-cluster $< V, A^0, ..., A^{k-2} >$, a set of switching points from appropriate windows of $V, A^0, ..., A^{k-2}$ will give rise to best and worst case coupling capacitances. When one is seeking for best case capacitance, rise/rise and fall/fall windows should be considered. For worst case coupling capacitance, rise/fall window of V and fall/rise window of $A^0, A^1, ..., A^{k-2}$ should be considered. A switching point is defined as an ordered pair of delay and slew. We consider maximum victim slew for worst case analysis . Minimum victim slew is considered for the best case.For the rest of the discussion we will consider worst case analysis as best case is symmetric to it. The following set gives all possible switching points in a k-cluster.

$$\{X^V, X^{A^0}, X^{A^1}, ..., X^{A^{k-2}}\} \tag{10}$$

Here X^j is the set of all switching point at node j where $j \in (V, A^0, A^1, ..., A^{k-2})$

THEOREM 1. *We find points $x \in X^V$ and $x^i \in X^{A^i}$ where $i \in (0, 1, ..., k-2)$ and form a set*

$$\{(x, x^0), (x, x^1), ..., (x, x^{k-2})\} \tag{11}$$

Equation 12 is a permutation of Equation 11

$$\{(x, \hat{x^0}), (x, \hat{x^1}), ..., (x, \hat{x^{k-2}})\} \tag{12}$$

Equation 12 forms a totally ordered set.

PROOF. We define a relation k-inclusion (that is $\sqsupseteq$)

$$k^{(x,x^i)} \geq k^{(x,x^j)} \Rightarrow (x, x^i) \sqsupseteq (x, x^j) \tag{13}$$

$$k^{(x,x^i)} < k^{(x,x^j)} \Rightarrow (x, x^j) \sqsupseteq (x, x^i) \tag{14}$$

where, $k^{(x,x^i)}$ is the overlap ratio between points x and x^i. By the definition of $\sqsupseteq$ and by the fact that a unique k exists between 0 and 1 for all 2-tuples (x, x^i),we can arrange Equation 11 in terms of increasing k. Equation 12 formed by the permutation of Equation 11 is a totally ordered set. □

According to lattice theory [6] a partially ordered set (which holds for a total order as well) forms a complete lattice if any subset has a least upper bound ($\top$) and greatest lower bound($\bot$) for its elements. Our set has a greatest lower bound when k is 0 and a least upper bound when k is 1 for all of its elements. Thus the set forms a complete lattice and if there is a order preserving transformation T [15], we can find the fix-points of this set using chaotic iteration scheme.

THEOREM 2. *Directed Search between a victim and an aggressor is an order preserving transformation. We call the transformation as DS*

$$\forall i : (x, x^i) \sqsupseteq (\tilde{x}, x^i) \Rightarrow DS(x, x^i) \sqsupseteq DS(\tilde{x}, x^i) \tag{15}$$

PROOF. Complete description of directed search is given in Section 3. Directed search between a victim and an aggressor come up with the worst possible coupling effect on the victim due to the aggressor and that changes the victim waveform by increasing the overlap ratio. Therefore

$$(x, x^i) \sqsupseteq (\tilde{x}, x^i) \Rightarrow k^{(x,x^i)} \geq k^{(\tilde{x},x^i)} \tag{16}$$

Algorithm: Worst-Coupling-Solver
Input: k-cluster $< V, A^0, ..., A^{k-2} >$
Output: Worst case coupling effect on V
begin
$C_c^{k-cluster} \leftarrow \sum_{i=0}^{k-2} C_c^i$
$C_c^{iter} \leftarrow 0$
Node-Queue$\leftarrow$ $A^0, A^1, ..., A^{k-2}$
while(Node-Queue is not empty)
$C_c^{iter} \leftarrow C_c^{k-cluster}$
$a \leftarrow$ Pop(Node-Queue)
Update $C_c^{k-cluster}$ due to $DS(V, a)$
if($C_c^{k-cluster} \neq C_c^{iter}$)
Push a into Node-Queue
endwhile
end

Figure 11: Worst Case Victim Coupling Capacitance Computation

By property of DS

$$DS(x, x^i) = (\hat{x}, x^i) \text{ where } k^{(\hat{x},x^i)} \geq k^{(x,x^i)} \quad (17)$$

$$DS(\tilde{x}, x^i) = (\bar{x}, x^i) \text{ where } k^{(\bar{x},x^i)} \geq k^{(\tilde{x},x^i)} \quad (18)$$

From Equations 16-18
$k^{(\hat{x},x^i)} \geq k^{(\tilde{x},x^i)} \Rightarrow DS(x, x^i) \sqsupseteq DS(\tilde{x}, x^i)$
□

We present a recursive formulation to apply the transformation DS on the set given by Equation 12

$$\bar{DS}\{(x, x^0), (x, x^1), ..., (x, x^{k-2})\} =$$
$$\bar{DS}(\bar{DS}\{(x, x^0), (x, x^1), ..., (x, x^{k-3})\}, \bar{DS}(x, x^{k-2}))$$
$$\bar{DS}(x, x^i) = DS(\hat{x}, x^i)$$

We always generate subsets of Equation 12 during the recursive application of DS. Chaotic iterations will reach to an optimal fixed point when

$$\bar{DS}(\{(\bar{x}, x^0), (\bar{x}, x^1), ..., (\bar{x}, x^{k-2})\}) = \quad (19)$$
$$\{(\bar{x}, x^0), (\bar{x}, x^1), ..., (\bar{x}, x^{k-2})\}$$

When the fixed point iteration converge, we get an effective capacitance which corresponds to the worst effect on victim due to all its aggressors. A pseudo-code of chaotic iteration scheme for worst case coupling capacitance calculation is shown in Figure 11. The algorithm presented in Figure 11 comes up with the worst case coupling capacitance on victim V due to all its aggressors. We call this capacitance as $C_c^{k-cluster}$. Similar formulation can be used to get the best case coupling capacitance.

5. ACCURATE ITERATIVE STATIC TIMER

We propose a chaotic iteration [15] based static timer algorithm in Figure 12 which makes use of k-cluster directed search to do accurate crosstalk aware timing analysis. All nodes in the graph G are topologically sorted and kept in a list N. N is the input to the non-iterative timer algorithm.

The algorithm initializes by putting all nodes N into a queue with $1C_c$ windows on each of them. The chaotic iteration proceeds as follows. A node from the queue is popped out and updated due to worst and best case coupling capacitances. Worst case coupling capacitance is obtained using Worst-Coupling-Solver presented in Figure 11 while

Algorithm: Accurate iterative static timer
Input: Topologically sorted Node list N
Output: Accurate Timing Windows on each Node
begin
for each node $v \in V$
Time v with 1C coupling capacitance
Node-Queue$\leftarrow$Node-Queue $\cup$ v
while(Node-Queue is not empty)
v $\leftarrow$Pop(Node-Queue)
Generate $k-cluster$ of v $(k-cluster^v)$
$C_{c_{worst}}^{k-cluster} \leftarrow$Worst-Coupling-Solver$(k-cluster^v)$
$C_{c_{best}}^{k-cluster} \leftarrow$Best-Coupling-Solver$(k-cluster^v)$
Time v with $C_{c_{worst}}^{k-cluster}$,$C_{c_{best}}^{k-cluster}$
Update C_c^v with $C_{c_{worst}}^{k-cluster}$,$C_{c_{best}}^{k-cluster}$
if(changes on windows of v > ϵ)
Add fan-outs of v to Node-Queue
Add aggressors of v to Node-Queue
endwhile
end

Figure 12: Accurate Static Timing Analysis Algorithm

best case coupling capacitance can be obtained by a routine similar to routine shown in Figure 11. If the timing windows on the victim changes, then all its aggressors are added uniquely to the queue. Victim capacitance is also updated to the best and worst case capacitance obtained. We reach the convergence if the windows do not defer by a tolerance ϵ.

5.1 Complexity

We present a brief sketch of complexity analysis of static timing analysis algorithm. Let the number of nodes in the graph G be N. [10] shows that complexity of bounded fixed point iteration can be quadratic in the worst case. Chaotic iterations from the outer loop of the algorithm has a complexity of $O(N^2)$. Suppose over all nodes, the maximum sized $k-cluster$ is composed of a m-tuple. Thus total aggressor nodes in the $k-cluster$ is (m-1). We can bound the complexity of Directed Search on k-cluster by $O((m-1)^2)$. Directed search operation takes $O(S)$ iterations where S represents the discrete values of k we want to consider between the upper and lower bounds [5] of k for performing Directed Search. Putting things together the total complexity of the algorithm can be given by $O(N^2 \cdot (m-1)^2 \cdot S)$. In practice fixed point iteration give linear complexity but what we showed in this section are strict bounds.

Instead of applying Directed Search to k-cluster of a node V, it can be applied to a local cluster [5] of V to get accurate capacitances and then iterations can be carried out on the basis of global cluster. In this paper although we focus on accuracy but directed search can be applied in a iterative framework as shown in [5] to get the best of both worlds.

6. EXPERIMENTAL RESULTS

6.1 Circuit modeling

We model a given circuit as a directed acyclic graph (referred to subsequently as the circuit's *timing graph*). We also introduce 2 virtual nodes PI and PO. PI is connected to all primary inputs of the circuit while all outputs are connected to PO. Nodes in the timing graph represent gates in the circuit while the edges represent the corresponding inter-

Table 1: Accuracy enhancement results

Circuit	# of Nodes	# of CE	$1C_c$ Window (rd_l, rd_h)(ns)	IST-(0,1,2) RT(s)	IST-(0,1,2) TA	IST-(0,1,2) (rd_l, rd_h)(ns)	IST-DS RT(s)	IST-DS TA	IST-DS (rd_l, rd_h)(ns)	%GA	%GT
c432	198	277	(0.48,15.72	0.06	1197	(0.02,24.43)	1.51	790	(0.40,35.21)	30.6	34.0
c499	245	312	(0.13,9.12)	0.06	1206	(0.07,26.93)	1.67	1005	(0.11,31.48	14.4	16.7
c880	445	620	(0.55,17.76)	0.23	4440	(0.16,26.98)	6.17	1786	(0.42,49.56)	45.5	59.8
c1355	589	829	(0.71,18.30)	0.30	6470	(0.20,31.01)	6.76	2915	(0.34,48.57)	36.2	54.9
c1908	915	1332	(0.55,27.05)	0.58	13009	(0.20,49.40)	22.5	6053	(0.27,72.51)	31.8	53.5
c2670	1428	1929	(0.01,24.03)	0.45	10094	(0.01,55.58)	40.34	6932	(0.02,61.18)	9.2	31.3
c3540	1721	2545	(0.32,33.10)	0.72	12822	(0.14,78.20)	87.80	11580	(0.32,87.07)	10.2	9.7
c5315	2487	3542	(0.15,30.49)	0.95	19578	(0.03,53.12)	142.00	12606	(0.15,77.93)	31.8	35.6
c6288	2450	3622	(0.70,81.54)	2.26	47135	(0.15,158.77)	198.97	16554	(0.57,194.03)	18.2	64.8
c7552	3721	5412	(0.01,31.87)	1.65	36887	(0.01.55.88)	325.86	21737	(0.01,77.63)	28.0	41.1

connects. We map all nodes to logic gates from the Faraday 90nm technology library. Delay models are available from look-up tables that yield a gate's delay and slew as functions of its input slew and load (output capacitance). Extracted coupling capacitance values are used to generate a *coupling graph* that denotes the timing dependencies introduced due to coupled nets. This coupling graph is then superimposed on the timing graph.

6.2 Results

We present accuracy enhancement results from accurate iterative static timer(IST-DS) on the ISCAS85 benchmarks [1] in Table 1. CE shows the number of coupling edges. k-cluster for each node is obtained and the node is timed with coupling capacitance $\sum_{i=0}^{k-2} C_c^i$. Timing result considering this situation is presented in $1C_c$ window. We present rise delay window (rd_l, rd_h). We compare our algorithm over a iterative algorithm based on the chaotic iteration theory presented in [15]. We call this algorithm as iterative static timer (IST-(0,1,2)) which considers a discrete coupling model with MCFs as 0,1,2 [12]. RT shows the runtime while TA shows the cell tables accesses. We show the Hold Time on the hypothetical node PO. Also we call worst case bound of timing windows as Hold Time to compare both approaches.

$$\%GA = \frac{\text{Hold Time}^{\text{IST-DS}} - \text{Hold Time}^{\text{IST-(0,1,2)}}}{\text{Hold Time}^{\text{IST-DS}}} \times 100.0$$

$$\%GT = \frac{\text{TA}^{\text{IST-(0,1,2)}} - \text{TA}^{\text{IST-DS}}}{\text{TA}^{\text{IST-(0,1,2)}}} \times 100.0$$

GA refers to the gain in accuracy and GT refers to gain in table access. Hold times given by IST are non-conservative compared to our algorithm which is a potential problem for the designer.Our algorithm increased the accuracy of crosstalk analysis on an average by 25.59% while on one particular circuit c880 the accuracy was improved by 45.5%. Also on an average our algorithm decreased the number of cell delay look-up table access by 40.1% whereas the maximum decrease we obtain is for circuit c6288 (64.8%). Cell table access is a costly operation in timing analysis and the search decreases it considerably. Note that this saving is not reflected in run time because of the complexity of directed search operation. Therefore directed search must be used judiciously to trade-off between speed and accuracy. Results shown are for experiments performed on a Pentium 2.4GHz processor server having 1Gb RAM and running RedHat Linux 9.0.

7. CONCLUSIONS AND FUTURE WORK

We present a predictive framework for accurate static timing analysis in UDSM VLSI circuits. A novel technique called directed search is developed to get accurate meaningful static timing and signal integrity analysis. An iterative static timing analysis algorithm is proposed which increases the accuracy of the analysis and also decreases the number of cell delay look-up table accesses. We also demonstrate accuracy enhancement by up-to 45.5% and decrease in cell delay look-up by up-to 64.8%. Directed search in itself is a costly operation. In the future we will study algorithms which apply the directed search selectively to get more accurate as well as efficient timing analysis.

8. REFERENCES

[1] F. Brglez and H. Fujiwara. A neutral netlist of 10 combinatorial benchmark circuits. In *ISCAS*, pages 695–698, 1985.

[2] P. Chen, D. A. Kirkpatrick, and K. Keutzer. Miller factor for gate-level coupling delay calculation. In *ICCAD*, San Jose, CA, Nov. 2000.

[3] P. Chen, Y. Kukimoto, and K. Keutzer. Refining switching window by time slots for crosstalk noise calculation. In *ICCAD*, pages 583–586, 2002.

[4] Faraday Technology Corporation. UMC 90-nano Libraries. `http://freelibrary.faraday-tech.com/`, Nov. 2005.

[5] D. Das, A. Shebaita, H. Zhou, Y. Ismail, and K. Killpack. FA-STAC: A framework for fast and accurate static timing analysis with coupling. In *ICCD*, 2006.

[6] B. A. Davey and H. A. Priestley. *Introduction to lattices and order*. Cambridge, 1990.

[7] M. Ghoneima and Y. Ismail. Accurate decoupling of coupled on-chip buses. In *ISCAS*, pages 4146–4149, 2005.

[8] A. Kahng, S. Muddu, and E. Sarto. On switch factor based analysis of coupled RC interconnects. In *DAC*, pages 79–84, 2000.

[9] R. Kumar. Interconnect and noise immunity design for the Pentium 4 processor. In *DAC*, pages 938–943, 2003.

[10] H. R. Nielson and F. Nielson. Bounded fixed point iteration. In *POPL*, pages 71–82, 1992.

[11] T. Sakurai and A. R. Newton. Alpha-Power Law MOSFET Model and its Applications to CMOS Inverter Delay and Other Formulas. *IEEE Journal of Solid state circuits*, 25:584–594, Apr. 1990.

[12] S. S. Sapatnekar. A timing model incorporating the effect of crosstalk on delay and its application to optimal channel routing. *IEEE TCAD*, 2000.

[13] Y. Shin and T. Sakurai. Coupling-Driven Bus Design for Low Power Application-Specific System. In *DAC*, pages 750–753, 2001.

[14] T. Xiao and M. Marek-Sadowska. Worst Delay Estimation in Crosstalk Aware Static Timing Analysis. In *ICCD*, 2000.

[15] H. Zhou. Timing analysis with crosstalk is a fixpoint on a complete lattice. In *IEEE Transactions on Computer-Aided Design, September 2003*, pages 1261–1269, 2003.