Crosstalk Analysis, Prevention and Repair Using PrimeTime-SI and Mars-Xtalk

Lars Bo Graversen
Jakob Salling
Rene Schaloffsky Bloch
Christian Folting

MIPS Technologies
larsg@mips.com

ABSTRACT

This paper will describe an evaluation flow for crosstalk analysis, repair and prevention using PrimeTime-SI and Mars-Xtalk. Results of the initial evaluation runs will be presented. The shortcomings of the flow will also be presented with suggestions for improvements.
1.0 Introduction

With the shrinking process geometries, the wire widths become smaller and the wires become taller and closer together. This has the effect of increasing the coupling capacitance between adjacent wires while decreasing the substrate capacitance. As the ratio of cross coupling capacitance to self-capacitance increases, the cross coupling capacitance can no longer be modelled as a simple capacitance (with a scaling factor) to ground.

Crosstalk is the effect of one net (the victim net) receiving undesirable cross-coupling from another net (the aggressor net) through the coupling capacitance between the two nets. Typically, a victim net has many aggressors, and an aggressor net is also a victim net. This obviously complicates the analysis somewhat.

For circuit geometries at 0.25 micron and above, substrate capacitance is typically the dominant effect and crosstalk has traditionally not been analyzed for designs in these technologies. Moving to 0.18 micron and below, the coupling capacitances become increasingly dominant and with this, the analysis, repair and prevention of crosstalk.

The effect of crosstalk may cause the victim net timing to slow down or speed up depending on the switching direction of the aggressor net with respect to the switching direction of the victim net. Slowdown of a data net (and/or speedup of a related clock net) may lead to setup violations and subsequently lower frequency of operation. This is not necessarily critical. Speedup of a data net (and/or slowdown of a related clock net) may lead to hold violations. This is critical as hold time violations will lead to non-functional silicon.

PrimeTime SI tends to add additional pessimism to the static timing analysis - and in some cases, probably too much pessimism (see also [2]). While it is of course a good thing that PrimeTime SI tends to be more pessimistic than for instance HSpice, as is reported in most cases (see [3], [4]), too much pessimism will lead to potentially signing off at a lower frequency than needed and/or attempting to overfix timing problems that are not real at the cost of area, speed and/or man-hours.

This paper describes evaluation work done at MIPS Technologies to implement an exemplary design flow containing features for crosstalk analysis, repair and prevention. Results from several designs of different complexity will be presented. Limitations to the flow and its tools will also be discussed.

2.0 Design Flows

The frontend part of the subject design flow uses DesignCompiler to perform bottom-up synthesis of RTL. From here, the design is taken through PhysicalCompiler in a gates-to-placed-gates flow. The resulting placement is taken through Apollo for clock tree synthesis and routing. This is followed by one or two IPO iterations through PhysicalCompiler (and back to Apollo) before final signoff.
2.1 Standard Flow
The relevant (for this paper) part of the standard, non-crosstalk aware, flow is illustrated in Figure 1.

![Figure 1. Standard flow](image)

The delay calculation and static timing analysis is based on SPEF backannotated from the extraction tool and crosstalk is not taken into consideration.
2.2 Crosstalk Aware Flow

Adding crosstalk analysis, repair and prevention to the flow in Figure 1. leads to a flow as illustrated in Figure 2.

![Crosstalk aware flow diagram]

2.3 Mars-Xtalk

Mars-Xtalk is an add-on to the Avant! (now Synopsys) Apollo router, and as such, the Apollo P&R flow is already prepared for using Mars-Xtalk. Very few and very simple modifications to the existing routing scripts are required to include crosstalk prevention and fixing during routing.

Mars-Xtalk provides three different methods for crosstalk prevention and fixing:

- The first two are very similar. They are prevention schemes that are used during global routing and track assignment (initial detailed routing). The main purpose is to avoid having nets with overlapping switching time windows assigned to the same global route cells (global routing) or assigned to adjacent tracks (track assignment). This will reduce the number of aggressor nets that are likely to contribute to the crosstalk induced delay on the victim net.
The final method is a fixing step which, based on crosstalk analysis, will try to fix the real
crosstalk problems in the layout. Initially, all nets are (by default) considered to be victims of
crosstalk. Up to three different pruning techniques can then be applied sequentially to filter out
those nets that are not suffering from crosstalk problems. The crosstalk analysis will generate
constraints for the remaining nets. These constraints are used to drive the search and repair
algorithm (final detailed routing), which will rip up these nets and reroute them according to the
constraints.

Having decided to use all three methods, the only parameter left to define is the threshold at which
a net will be considered a victim of crosstalk. By default, the threshold is 0.45, meaning that a net
is considered to have a crosstalk violation if one or more aggressors induce a total noise voltage
greater than 0.45*VDD on the net in question.

Mars-Xtalk does not consider the timing effects of crosstalk. The timing windows used to
determine whether a signal is an aggressor and/or victim with respect to another signal are
calculated without taking crosstalk into account. This means that some relevant aggressor/victim
pairs can be missed. The delay or speedup effect on a victim net is not calculated either. Noise
voltage level is the only criteria for selection of nets for analysis and repair (based on static timing
windows).

2.4 PrimeTime-SI
PrimeTime-SI (PT-SI) is an optional enhancement to the PrimeTime (PT) static timing analysis
(STA) tool. It adds crosstalk analysis capabilities to the PrimeTime static timing analysis engine,
[5]. As an enhancement to (or super set of) PrimeTime, it is very easy to setup and integrate with
an existing PT based STA flow.

To get started with the default variable settings, a parasitic backannotation file containing cross
coupling capacitances is needed. The format for this file is IEEE-1481 Standard Parasitics
Exchange Format (SPEF). The backannotation file with cross coupling capacitances can be read
using the following command:

```
read_parasitics -keep_coupling_capacitances <file.spef>
```

The following commands must be issued before `update_timing` to enable crosstalk analysis:

```
set_operating_conditions -analysis_type on_chip_variation
set si_enable_analysis true
```

The first command switches the analysis type to `on_chip_variation` which is how PT-SI
handles the min-max timing window relationships needed for crosstalk analysis. Inherently, this
means that simultaneous min-max analysis using commands like "set_operating_conditions
-min BCCOM -max WCCOM -analysis_type bc_wc" is not supported in PT-SI. If this analysis
type is not specified explicitly, PT-SI automatically switches to `on_chip_variation`. Setting
`si_enable_analysis` to `true` enables the crosstalk analysis.
With this setup, PrimeTime-SI performs crosstalk analysis using the default values for the remaining relevant variables, see [5]. Crosstalk analysis is an iterative process. PT-SI selects initial nets to be considered possible aggressors and/or victims based on parasitic and electrical filtering. For the initial analysis of these nets, PT-SI assumes that every aggressor can have a worst type transition (rising or falling) at the worst possible time. This will cause the worst possible speedup or slowdown of transitions on the victim net. In the second (and subsequent) delay calculation iterations, the possible switching times of aggressors and victims are considered and crosstalk delays that can never occur (based on the separation in time between aggressor and victim switching) are disregarded. After each delay calculation iteration, PT-SI selects a new set of nets for analysis in the next iteration. After reselection, it is determined whether enough iterations have been performed based on the exit criteria.

The variable controlling the exit criteria in the exemplary flow is set as follows (except where otherwise noted):

\[
\text{set si_xtalk\_exit\_on\_max\_iteration\_count 2}
\]

which is the default value. This means that PT-SI exits from the analysis loop after two iterations. A value of 3 was also tested, but the additional iteration did not change the results significantly.

The parasitic filtering variables are set as follows (except where otherwise noted):

\[
\begin{align*}
\text{set si\_filter\_total\_aggr\_xcap 0.025} \\
\text{set si\_filter\_total\_aggr\_xcap\_to\_gcap\_ratio 0.025}
\end{align*}
\]

This means that a net is disregarded as a potential victim if the total crosscoupling capacitance between the potential victim net and all aggressor nets is less than 0.025 library units and the ratio of the total crosscoupling capacitance between the potential victim net and all aggressor nets to the total ground capacitance of all subnodes of the potential victim net is less than 0.025.

The variables controlling which nets are reselected for further analysis are set as follows (except where otherwise noted):

\[
\begin{align*}
\text{set si\_xtalk\_reselect\_critical\_path false} \\
\text{set si\_xtalk\_reselect\_delta\_delay 0.1} \\
\text{set si\_xtalk\_reselect\_delta\_delay\_ratio 1000} \\
\text{set si\_xtalk\_reselect\_max\_mode\_slack -0.1} \\
\text{set si\_xtalk\_reselect\_min\_mode\_slack -0.1}
\end{align*}
\]

With \text{si\_xtalk\_reselect\_critical\_path} set to true, only nets in the critical path for each path group are selected for further crosstalk analysis. Setting it to false means that the other variables control which nets are reselected for further analysis.

The setting of \text{si\_xtalk\_reselect\_delta\_delay} means that all nets with a crosstalk delta delay of more than 0.1ns will always be reselected. This variable also controls the reselection of clock nets. Clock nets are not reselected based on the *\_min/max\_mode\_slack variables.
The settings of `si_xtalk_reselect_max_mode_slack` and `si_xtalk_reselect_min_mode_slack` means that all nets with a net slack of less than -0.1ns will be reselected. This means that nets on paths with a timing violation of more than 0.1ns are reselected. This is done for initial runs to avoid too many nets being reselected (the number of reselected nets has a great impact on the runtime of `update_timing`). For final (signoff) runs, it is preferred to set

```
set si_xtalk_reselect_max_mode_slack 0
set si_xtalk_reselect_min_mode_slack 0
```

to ensure that all nets on violating paths are reselected.

### 3.0 Design #1

This design is a small block without memories or other large macros. The design has the following features:

- Technology: TSMC 0.13 LV-LowK
- 18K placeable instances (4K flops)
- 20K nets
- Layout size approximately 600µ by 600µ
- No clock gating

Several experiments were run with this design, trying different threshold values for Mars-Xtalk. The first run, ‘Exp_1’ serves as the reference. It represents the original P&R flow without Mars-Xtalk. The results are summarized in Table 1. The ‘t’ value indicated in the table is the threshold value for Mars-Xtalk, indicating the voltage bump level considered as being a crosstalk problem by the tool. All operating frequencies in the table are normalized with the PrimeTime (non-SI) analysis result before any crosstalk repair and prevention being 100.

<table>
<thead>
<tr>
<th>Run</th>
<th>Runtime Global route</th>
<th>Runtime Detail route</th>
<th>Relative performance PrimeTime</th>
<th>Relative performance PrimeTime-SI</th>
</tr>
</thead>
<tbody>
<tr>
<td>Exp_1</td>
<td>7 min.</td>
<td>36 min.</td>
<td>100</td>
<td>89.2</td>
</tr>
<tr>
<td>Exp_2 (t=0.45)</td>
<td>9 min.</td>
<td>65 min.</td>
<td>n/a</td>
<td>93.5</td>
</tr>
<tr>
<td>Exp_3 (t=0.30)</td>
<td>9 min.</td>
<td>78 min.</td>
<td>n/a</td>
<td>97.9</td>
</tr>
<tr>
<td>Exp_4 (t=0.25)</td>
<td>9 min.</td>
<td>177 min. (with DRC errors)</td>
<td>n/a</td>
<td>92.5</td>
</tr>
</tbody>
</table>
The relevant PT-SI variables were set as follows for these runs:

\[
\begin{align*}
\text{si_filter_total_aggr_xcap} & = 0.025 \\
\text{si_filter_total_aggr_xcap_to_gcap_ratio} & = 0.025 \\
\text{si_xtalk_exit_on_max_iteration_count} & = 3 \\
\text{si_xtalk_reselect_critical_path} & = \text{false} \\
\text{si_xtalk_reselect_delta_delay} & = 5 \\
\text{si_xtalk_reselect_delta_delay_ratio} & = 0.95 \\
\text{si_xtalk_reselect_max_mode_slack} & = 0 \\
\text{si_xtalk_reselect_min_mode_slack} & = 0
\end{align*}
\]

Statistics for PrimeTime-SI run:

- PrimeTime-SI version used: 2002.03-Beta1
- Memory usage: 152 MB
- CPU time: 19 min.
- Machine used: SunBlade 1000, 750MHz, 4 GB physical memory

From Exp_1, it can be seen that the original P&R flow without crosstalk awareness in the router has a considerable performance degradation (10.8\%) due to crosstalk when analyzed using PrimeTime-SI.

It is clear from the above results that there is an optimal value of the threshold for considering nets as having crosstalk problems. In this example, this optimal value is around \(t=0.30\). This value is probably very design dependent, depending (among other things) on the congestion of the routing. The more dense the routing, the more difficult it is to repair crosstalk problems. Selecting too many nets for repair will make it impossible for the router to complete and fix the problems. This happens somewhere between \(t=0.20\) and \(t=0.25\) for this design.

In this case (\(t=0.30\)), Mars-Xtalk seems to perform rather well by reducing the performance degradation to approximately 2.1\%. The penalty is approximately 2x in runtime.

### 3.1 An Improved Crosstalk Repair Flow

The flow used above has a fundamental problem. While PrimeTime-SI analyzes the degradation of the path delays due to the crosstalk effects, Mars-Xtalk is only concerned about the noise voltages induced from one wire to another. Potential victim nets of crosstalk are selected based...
only on the size of the noise voltage induced by nearby aggressors (taking timing windows into account, though). Victim nets receiving a noise voltage smaller than the specified threshold value, will not be considered during the subsequent crosstalk repair.

The problem is, that even a sub-threshold noise voltage could result in a delay large enough to create a setup violation or a negative delay large enough to create a hold-time violation. Especially if the path was already close to the limit timing-wise. Mars-Xtalk does not look at the overall timing aspect. The timing engine is not capable of handling crosstalk induced timing delays.

It means, that Mars-Xtalk is only indirectly solving the crosstalk problems, and that you have to overconstrain to get the best possible result. It is obvious from the results in Table 1, that it is not feasible to place crosstalk constraints on too may nets at the same time. The algorithm will not be able to complete the routing in reasonable time and without DRC errors.

In order to overcome the missing connection between crosstalk analysis and crosstalk repair, and to catch the nets that are potentially missed (for the reasons explained above), another iteration to the flow was added as illustrated in figure 3.

Figure 3. Improved crosstalk aware flow.
During crosstalk and timing analysis, PrimeTime-SI provides a list of victim nets which are subject to crosstalk induced delays above a certain (relatively low) limit, and a list of nets which are in the critical timing paths. If a net is present in both lists, then it is both a potential timing violator and a victim of crosstalk, and it is possible that removing the crosstalk induced delay from this net will also improve the overall path timing delay, thus decreasing the timing violation.

Such nets are chosen for the next iteration of repair and a new script is generated for the router, which contains additional, and much tighter, crosstalk constraints for the chosen nets. Despite the harder constraints, this is still a manageable job for the router, because the number of ‘chosen nets’ is significantly smaller than the original number of nets processed by Apollo/Mars-Xtalk.

The new flow iteration was implemented using a Perl script as the “Crosstalk Constraint Generator”.

With this new methodology in place, the first round of crosstalk analysis and repair, which is done by Apollo/Mars-Xtalk, is now reduced to a matter of keeping the noise level below a certain specified limit for the entire design in general. A general suppression of the noise peaks to approximately 0.5 times VDD is usually wanted to assure that no false triggering occurs, e.g. of reset signals and interrupts. The more advanced and accurate crosstalk analysis by PT-SI and the post-processing of its reports by the “Crosstalk Constraint Generator” will then decide which nets to focus on in the second repair step, in order to improve the overall timing. In this way, Apollo/Mars-Xtalk is used for what it was designed to do (and does pretty well) and a link between analysis and repair has been established.

The new methodology was tested on the same design as referenced in Table 1. The new results are shown in Table 2.

<table>
<thead>
<tr>
<th>Run</th>
<th>Runtime Global route</th>
<th>Runtime Detail route</th>
<th>Runtime Additional iteration</th>
<th>Relative performance PrimeTime-SI</th>
</tr>
</thead>
<tbody>
<tr>
<td>Exp_6</td>
<td>9 min.</td>
<td>78 min.</td>
<td>66 min.</td>
<td>98.1</td>
</tr>
</tbody>
</table>

In this case, only slightly improved timing was observed. One reason for this is, that the nets (path segments) that are victims of crosstalk, are not at the same time the most timing critical segments of the violating timing path. Thus, reducing the crosstalk induced delay on those segments, did not affect the overall timing of the entire path greatly.

Another reason could be, that by focussing on fixing crosstalk problems for a particular set of nets, Mars-Xtalk might actually increase the crosstalk effect on other nets, which were not critical initially. To analyze this in detail is a huge and time consuming task, which was not pursued.
4.0 Design #2

This design is a larger core, including some memories, implemented in TSMC 0.13G.

For this design, the SPEF format backannotation file including cross coupling capacitances is rather large. Some statistics on backannotation for this design are shown in Table 3. Synopsys Binary Parasitics Format (SBPF) is a new (binary) format that can be written out from PT-SI after reading in a SPEF file and read back in for subsequent timing analyses. The advantages of the format are obvious. Not only is the file size much smaller, but it also loads orders of magnitude faster.

<table>
<thead>
<tr>
<th>File format</th>
<th>File size</th>
<th>Time to load</th>
</tr>
</thead>
<tbody>
<tr>
<td>IEEE-1481 SPEF</td>
<td>380 MB</td>
<td>103 min.</td>
</tr>
<tr>
<td>Synopsys Binary Parasitics Format (SBPF)</td>
<td>44.2 MB</td>
<td>&lt; 1 min.</td>
</tr>
</tbody>
</table>

With this design, several experiments have been carried out trying to improve the crosstalk performance. The basic flow setup is as described above for design #1, but constraints for physical synthesis (using PhysicalCompiler) have been modified for the experiments, trying to improve the final result by generating a better input netlist to the initial routing.

As PhysicalCompiler (PC) has no concept of crosstalk (or any other signal integrity issues), it is not possible to optimize the design for crosstalk performance directly. The idea behind the experiments was to limit crosstalk problems by reducing the sensitive timing windows of nets. The key to achieving this is to tighten the max transition time constraint for all nets in the design. The baseline PC run used a max_transition constraint of 0.8 ns and a max_fanout constraint of 20. It was noted that tightening the max_transition to 0.3 ns while keeping the max_fanout constraint at 20 led to an increase in area of the initial placed netlist of 5.4% while the worst negative slack (WNS) increased by 0.2 ns. Also tightening the max_fanout constraint to 4 led to an increase in area by 2.4% while WNS improved by 0.1 ns. It was decided to use this setting (max_transition = 0.3 ns, max_fanout = 4) moving forward.

As a side effect, it was noted that small changes in certain routing parameters (congestionWeight and XtalkWeight) in Apollo/Mars-Xtalk helped to further reduce the number of crosstalk sensitive nets. Double spacing to clock nets at the higher levels of the clock tree was also introduced during routing to further decrease the crosstalk sensitivity of the clock nets.

Results from some of the runs are shown in Table 4. Exp_1 is the baseline run. The performance of this design without taking crosstalk induced delays into consideration is set to be 100 for reference. All other timing numbers are relative to this. Each run has been through a flow similar to the one illustrated in Figure 2. Two IPO iterations were performed for each. The “Flow”
column in Table 4 indicates the router configuration for each of the three runs through the router. These settings are detailed in Table 5. The “# of xtalk max violations” column shows the total number of max timing violations caused by crosstalk delta delays for each run.

Table 4: Results, design #2

<table>
<thead>
<tr>
<th>Run</th>
<th>max_transition/ max_fanout</th>
<th># of xtalk max violations</th>
<th>Flow</th>
<th>Relative perf. PrimeTime</th>
<th>Relative perf. PT-SI</th>
</tr>
</thead>
<tbody>
<tr>
<td>Exp_1</td>
<td>0.8/20</td>
<td>452</td>
<td>c1/c1/c1</td>
<td>100</td>
<td>94.3</td>
</tr>
<tr>
<td>Exp_2</td>
<td>0.3/4</td>
<td>423</td>
<td>c1/c1/c1</td>
<td>100</td>
<td>91.9</td>
</tr>
<tr>
<td>Exp_3</td>
<td>0.3/4</td>
<td>254</td>
<td>c1/c2/c3</td>
<td>94.1</td>
<td>91.4</td>
</tr>
<tr>
<td>Exp_4</td>
<td>0.3/4</td>
<td>213</td>
<td>c3/c3/c3</td>
<td>91.6</td>
<td>89.2</td>
</tr>
</tbody>
</table>

Table 5: Details of routing flow setup

<table>
<thead>
<tr>
<th>Parameters &amp; procedures</th>
<th>Flow</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>c1</td>
</tr>
<tr>
<td>congestionWeight</td>
<td>3</td>
</tr>
<tr>
<td>XtalkWeight</td>
<td>4</td>
</tr>
<tr>
<td>Double spacing to clock nets</td>
<td>No</td>
</tr>
</tbody>
</table>

The (relevant) PT-SI variable settings for these runs were as follows:

\[
\text{si}_\text{_xtalk_reselect_delta_delay} = 0.005 \\
\text{si}_\text{xtalk_reselect_max_mode_slack} = 0 \\
\text{si}_\text{xtalk_reselect_min_mode_slack} = 0
\]

Among the runs summarized above, there is no clear winner. The initial run (Exp_1) has the best performance, while Exp_3 and Exp_4 have slightly worse performance, but much smaller degradation due to crosstalk - and fewer violations. This is work in progress, no best setup or flow has been identified yet.
5.0 Design #3

This design is implemented in TSMC 0.18G. It is included here as an example of crosstalk fixing in the described flow. This work was done with PrimeTime-SI version 2002.03-SP1.

An example of a timing report from PT-SI before using the crosstalk prevention/repair flow is shown below. Hierarchy has been removed from this (and the following) report for simplicity.

Startpoint: register
(rising edge-triggered flip-flop clocked by SI_ClkIn)

Endpoint: RAM
(rising edge-triggered flip-flop clocked by SI_ClkIn)

Path Group: SI_ClkIn
Path Type: max

<table>
<thead>
<tr>
<th>Point</th>
<th>Fanout</th>
<th>DTrans</th>
<th>Trans</th>
<th>Delta</th>
<th>Incr</th>
<th>Path</th>
</tr>
</thead>
<tbody>
<tr>
<td>clock SI_ClkIn (rise edge)</td>
<td>0.0000</td>
<td>0.0000</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>clock network delay (propagated)</td>
<td>2.5079</td>
<td>2.5079</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>register/CK (SDFFX2)</td>
<td>0.2750</td>
<td>0.0000</td>
<td>2.5079 r</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>register/Q (SDFFX2)</td>
<td>0.2677</td>
<td>0.4534</td>
<td>2.9612 r</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>n2600 (net)</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>U315/A (INVX1)</td>
<td>0.0844</td>
<td>0.3520</td>
<td>2.5079</td>
<td>0.1016 f</td>
<td></td>
<td></td>
</tr>
<tr>
<td>U315/Y (INVX1)</td>
<td>0.3482</td>
<td>0.2485</td>
<td>3.3113 f</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>n2599 (net)</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>U316/A (INVX1)</td>
<td>0.3229</td>
<td>0.6711</td>
<td>0.1907</td>
<td>0.1928 f</td>
<td></td>
<td></td>
</tr>
<tr>
<td>U316/Y (INVX1)</td>
<td>0.2079</td>
<td>0.1706</td>
<td>3.6747 r</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>n2599 (net)</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>U324/A (NAND2X1)</td>
<td>0.0000</td>
<td>0.2079</td>
<td>0.1907</td>
<td>0.1928 f</td>
<td></td>
<td></td>
</tr>
<tr>
<td>U324/Y (NAND2X1)</td>
<td>0.1322</td>
<td>0.0810</td>
<td>3.5041 f</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>data_bit_27 (net)</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>data_bit_27</td>
<td>0.0000</td>
<td>0.0000</td>
<td>4.0946 r</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>n4270 (net)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>U112/A (BUFX8)</td>
<td>0.0533</td>
<td>0.7049</td>
<td>0.1460</td>
<td>0.1475 f</td>
<td></td>
<td></td>
</tr>
<tr>
<td>U112/Y (BUFX8)</td>
<td>0.6039</td>
<td>0.3388</td>
<td>4.6373 r</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RAM/D[27]</td>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RAM/D[27]</td>
<td>0.3250</td>
<td>0.1121</td>
<td>0.3304</td>
<td>5.0100 r</td>
<td></td>
<td></td>
</tr>
<tr>
<td>data arrival time</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>data required time</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Highlighted in the above timing report are three rather large crosstalk delta delays. The largest of
these (the net connected to RAM/D[27]) has crosstalk bump sizes reported by PT-SI as follows:

- Total bump rise: 0.4022
- Total bump fall: 0.4904

The bump sizes are reported as a percentage of VDD.
After going through the crosstalk flow, the path is as follows:

Startpoint: register
   (rising edge-triggered flip-flop clocked by SI_ClkIn)
Endpoint: RAM
   (rising edge-triggered flip-flop clocked by SI_ClkIn)
Path Group: SI_ClkIn
Path Type: max

<table>
<thead>
<tr>
<th>Point</th>
<th>Fanout</th>
<th>DTrans</th>
<th>Trans</th>
<th>Delta</th>
<th>Incr</th>
<th>Path</th>
</tr>
</thead>
<tbody>
<tr>
<td>clock SI_ClkIn (rise edge)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>clock network delay (propagated)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>register/CK (SDFFX4)</td>
<td>0.3761</td>
<td></td>
<td></td>
<td>0.0000</td>
<td>2.7179</td>
<td></td>
</tr>
<tr>
<td>register/Q (SDFFX4)</td>
<td>0.2536</td>
<td></td>
<td></td>
<td>0.4104</td>
<td>3.1284</td>
<td></td>
</tr>
<tr>
<td>data_bit_27 (net)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>U209/B (NAND2X1)</td>
<td>0.0638</td>
<td>0.3177</td>
<td>0.0489</td>
<td>0.0535</td>
<td>3.1819</td>
<td>r</td>
</tr>
<tr>
<td>U209/Y (NAND2X1)</td>
<td>0.2460</td>
<td></td>
<td></td>
<td>0.1635</td>
<td>3.3454</td>
<td>f</td>
</tr>
<tr>
<td>n3652 (net)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>U324/A (NAND2X4)</td>
<td>0.0000</td>
<td>0.2460</td>
<td>0.0000</td>
<td>0.0003</td>
<td>3.3458</td>
<td>f</td>
</tr>
<tr>
<td>U324/Y (NAND2X4)</td>
<td>0.1351</td>
<td></td>
<td></td>
<td>0.1009</td>
<td>3.4466</td>
<td>r</td>
</tr>
<tr>
<td>n18910 (net)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>U427/A (INVX1)</td>
<td>0.0997</td>
<td>0.1447</td>
<td>0.0081</td>
<td>0.0084</td>
<td>3.4550</td>
<td>r</td>
</tr>
<tr>
<td>U427/Y (INVX1)</td>
<td>0.1755</td>
<td></td>
<td></td>
<td>0.1255</td>
<td>3.5805</td>
<td>f</td>
</tr>
<tr>
<td>n18909 (net)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>U428/A (BUFX1)</td>
<td>0.0525</td>
<td>0.2281</td>
<td>0.0219</td>
<td>0.0225</td>
<td>3.6030</td>
<td>f</td>
</tr>
<tr>
<td>U428/Y (BUFX1)</td>
<td>0.1700</td>
<td></td>
<td></td>
<td>0.1177</td>
<td>3.7207</td>
<td>r</td>
</tr>
<tr>
<td>n18911 (net)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>U426/A (BUFX3)</td>
<td>0.0000</td>
<td>0.1282</td>
<td>0.0000</td>
<td>0.0000</td>
<td>3.8666</td>
<td>r</td>
</tr>
<tr>
<td>U426/Y (BUFX3)</td>
<td>0.1526</td>
<td></td>
<td></td>
<td>0.1572</td>
<td>4.0239</td>
<td>r</td>
</tr>
<tr>
<td>n4270 (net)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>U112/A (BUFX16)</td>
<td>0.0000</td>
<td>0.1526</td>
<td>0.0000</td>
<td>0.0004</td>
<td>4.0243</td>
<td>r</td>
</tr>
<tr>
<td>U112/Y (BUFX16)</td>
<td>0.3132</td>
<td></td>
<td></td>
<td>0.1949</td>
<td>4.2191</td>
<td>r</td>
</tr>
<tr>
<td>data_bit_27 (net)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RAM/D[27]</td>
<td>0.0497</td>
<td>0.2607</td>
<td>0.0412</td>
<td>0.0752</td>
<td>4.2944</td>
<td>r</td>
</tr>
<tr>
<td>data arrival time</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>clock SI_ClkIn (rise edge)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>clock network delay (propagated)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RAM/CLK</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>library setup time</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>data required time</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>slack (MET)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

and the crosstalk bump sizes on the net connected to RAM/D[27] are as follows:

Total bump rise: 0.1789
Total bump fall: 0.1699
The bump sizes and crosstalk deltas are reduced significantly in this case.

The crosstalk has been reduced by several changes in the path:

1. The startpoint flop has been upsized
2. Two inverters on the startpoint flop output have been removed
3. The second NAND gate has been upsized
4. One buffer has been replaced by a chain of two inverters and two buffers
5. Most likely some routing changes, although this cannot be verified by looking at the timing report alone

The routing changes have been performed by Mars-Xtalk while the buffering changes and upsizing are a result of the in-place optimization (IPO) performed in PhysicalCompiler.

The IPO is based on backannotated information from the routed design, but apart from this, it is not crosstalk aware as such, so the crosstalk improvements are a side effect of the normal IPO run. In the crosstalk flow, the max_transition constraint was also tightened (when compared to the original run) to attempt to force buffering changes which would improve the crosstalk performance. For this particular path, this seems to work, but in general it is probably not guaranteed to be the case. While fast transition times are desirable for crosstalk victims, it also increases the same nets effect as an aggressor. Without the IPO tool (in this case PhysicalCompiler) being crosstalk aware, it may not be deterministic whether the buffering changes caused by tighter max_transition constraints actually improve the overall crosstalk performance of the design. The flow is not guaranteed to converge.

6.0 Clock Net Considerations

Obviously, clock nets can be victims as well as aggressors with respect to crosstalk induced delays. This is taken into consideration in PT-SI crosstalk delay calculations. There are a few issues to consider in these calculations, however.
6.1 Clock Tree Delta Delay Calculations

Consider a clock tree like the one illustrated in Figure 4. The buffers B1 and B2 are common to the clock trees for the flip-flops FF1 and FF2, while B3A and B4A are unique to FF1 and B3B and B4B are unique to FF2.

In case of a maximum (setup) timing check, timing is related to two consecutive clock edges, the first clock edge launching the data from the startpoint flop (FF1) and the subsequent clock edge capturing data at the endpoint flop (FF2). In this case, the worst case timing is achieved with the slowest possible clock at FF1 and the fastest possible clock at FF2. As two different clock edges are involved in this calculation, it is relevant to consider the full crosstalk slowdown delta from CLK through B1, B2, B3A and B4A to the clock pin of FF1 for the startpoint calculation and the full crosstalk speedup delta from CLK through B1, B2, B3B and B4B to the clock pin of FF2 for the endpoint calculation. This is the default behavior for PT-SI.

For minimum (hold) timing checks, the timing is related to the same clock edge from CLK to FF1 and FF2 respectively. The worst case timing for hold checks is achieved with the fastest possible clock at FF1 and the slowest possible clock at FF2. In this case, only one clock edge from CLK is involved, so the speedup to the clock pin of FF1 should only include the crosstalk speedup delta from the output of B2 through B3A and B3B to FF1, and the slowdown to the clock pin of FF2 should only include the slowdown from the output of B2 through B3B and B4B to FF2. The speedup or slowdown from CLK through B1 to the input pin of B2 is common to the two paths and should thus be disregarded from the calculation. This is NOT the default behavior of PT-SI version 2002.03-SP1 (or earlier). By default, PT-SI uses the full speedup from CLK through B1, B2, B3A and B4A to FF1 for the startpoint calculation and the full slowdown from CLK through B1, B2, B3B and B4B to FF2 for the endpoint calculation. This is unnecessarily pessimistic as it assumes that the same clock edge can be simultaneously sped up and slowed down by crosstalk effects from CLK through B1 to B2. This is not possible. It leads to a (sometimes) large number of non-existent hold time errors being reported.
A suggested workaround for this (see also [2]) is to use clock reconvergence pessimism removal (CRPR). This feature is enabled by setting the variable `timing_remove_clock_reconvergence_pessimism` to `true`. This seems to work in most cases for flop to flop minimum (hold) timing checks, correctly adjusting the timing reports to compensate for the delays in the common part of the clock trees, but it does have some side effects.

With CRPR enabled, PT-SI also removes the difference in delays between common parts of the clock tree for setup checks. This should not be done for crosstalk enabled timing analysis, as it is possible for the first clock edge to be slowed down and the second to be sped up by crosstalk effects through the common part of the clock tree. Removing the difference in delay through the common part of the clock tree leads to setup timing reports to be too optimistic with CRPR and crosstalk analysis enabled. Removing the common path delay difference for setup checks is the correct behavior for on-chip variance timing analysis, which is what CPRP is designed for, but it is NOT correct for crosstalk enabled timing analysis. Synopsys is aware of this problem and has committed to fix it in a (not too distant) future version of PT-SI.

### 6.2 Clock Gating Circuit

Consider the discrete clock gating circuit illustrated in Figure 5. In this circuit, clock gating checks are defined on the input pins of the clock gating AND gate. The clock gating setup time \( t_{gc,\text{setup}} \) is defined such that the condition signal COND must arrive at the A input of the clock gate no later than \( t_{gc,\text{setup}} \) before the rising edge of CLK arrives at B. Similarly, the clock gating hold time \( t_{gc,\text{hold}} \) is defined such that the COND should be stable at A at least \( t_{gc,\text{hold}} \) after the falling edge of CLK arrives at B. For the hold check, it is the same CLK clock edge that enables the latch and is being gated at the clock gate. The only crosstalk delta delay that should be taken into consideration for this calculation is the speedup on net segment 1 and the slowdown on net segment 2. PT-SI does not handle this correctly, even with CRPR enabled. With CRPR enabled, PT-SI (version 2002.03-SP1 and earlier) compensates some of the delay through the common part of the clock tree, but the compensation is not calculated correctly and it is not immediately obvious how the number output in the timing reports is derived. Synopsys is aware of this problem as well, and a (not too distant) future version of PT-SI will (hopefully) have a correct calculation for this timing check as well as a feature to report details on how the CRPR compensation in the timing reports has been calculated.
Finally, in some cases (on designs with many clock gating checks in particular), turning on CRPR increases the memory usage and runtime of update_timing by a factor of more than 3x. This has been improved somewhat with the latest PT-SI release 2002.03-SP1, limiting the memory usage increase to approximately 1.5x for one of our test cases.

In general, it must be concluded, that CRPR is not designed for use with crosstalk enabled timing analysis. The simple and obvious rule should be that the same edge travelling through the same path/cells should always have the same delay (no crosstalk delta) for startpoint and endpoint delays while different edges travelling through the same path/cells should always have different delays (worst case crosstalk delta) for startpoint and endpoint delays. CRPR does not deliver this.

Bearing this in mind, it has been decided not to rely on CRPR in the flow at this time. All timing reports are generated without CRPR. This currently generates the most realistic setup checks and too pessimistic hold checks. A perl script is then used to compensate for the additional pessimism in the minimum timing reports in estimating whether any reported hold time violations are real.

6.3 Reselection Variable Settings
The pessimism of timing calculation of clock nets (as well as other nets) is influenced by the setting of the variable si_xtalk_reselect_delta_delay. The default value for this variable is 5.0, meaning that only nets with a crosstalk delta delay larger than 5.0ns are reselected for the next iteration (if the nets are not reselected for other reasons). This variable also influences the reselection of clock nets (unlike the variables si_xtalk_reselect_max_mode_slack and si_xtalk_reselect_min_mode_slack). Keeping this variable at the default setting or setting it at a lower (but still too high) value could mean that clock nets on paths violating hold (and/or setup) checks are not reselected. This means that the nets keep the crosstalk delta delays from the first iteration. This iteration assumes infinite switching windows for victims and aggressors which leads to too pessimistic crosstalk delta delays, potentially causing hold time calculations to be more pessimistic than needed. Whether or not this is critical or significant is design dependent.
For one of the designs used in our evaluation, the effect of this variable on overall hold violations is illustrated in Table 6. The “total hold violations” is taken from the \texttt{min\_delay/hold} line of the output of \texttt{report\_constraint}.

<table>
<thead>
<tr>
<th>si_xtalk_reselect_delta_delay setting</th>
<th>Total hold violations</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1</td>
<td>35.9856</td>
</tr>
<tr>
<td>0.02</td>
<td>9.2775</td>
</tr>
<tr>
<td>0.005</td>
<td>6.8672</td>
</tr>
</tbody>
</table>

The optimum setting for \texttt{si_xtalk_reselect_delta_delay} is design dependent. Setting it too high may lead to many false hold violations being reported, while setting it lower leads to many additional nets (clock as well as data nets) being reselected resulting in an increase in runtime (the runtime for an iteration is more or less proportional with the number of nets being evaluated).

### 7.0 Next Steps

The possible next steps in the evaluation of crosstalk aware design tools and flows will focus on (among other things) the following issues:

- Automation of prevention and repair in a timing based flow. The experience with the current flow shows that it is essential for good and reliable results, that the prevention and repair tool understands the timing consequences of crosstalk.

- Hold time errors need to be fixed and it is essential to understand and trust the hold time calculations under the influence of crosstalk induced delays (and speedups). Using clock reconvergence pessimism removal as part of the PT-SI delay calculation, it can be very difficult to verify that the reported hold time violations (or lack thereof) are real. As hold time errors in final silicon are critical, better reporting of these is needed.

### 8.0 Conclusions and Recommendations

From the analyses reported in this paper, it is clear that crosstalk induced delay is an issue which must be considered for 0.13 micron designs and maybe even for 0.18 micron designs. It is also clear, that a flow to perform crosstalk analysis, prevention and repair can be built around PrimeTime-SI and Mars-Xtalk, but requires some scripting.

The flow evaluated in this paper must be considered a first generation crosstalk aware flow. There are some obvious flaws in the flow, primarily caused by tool limitations. It is a major convergence problem that the prevention/repair tool and the analysis tool have different ways of calculating the crosstalk influence on the design. Ideally, the repair tool should have the full analysis capabilities.
Mars-Xtalk considers only on the size of the voltage bumps induced on a victim net by one or more aggressor nets. This ignores the fact that even a small voltage bump may cause a timing violation on a critical path while a larger voltage bump on a non-critical path may not matter.

PrimeTime-SI on the other hand is able to calculate the timing consequences of crosstalk induced delays with some degree of accuracy, for setup checks in particular, but is not able to directly guide or drive the routing tool during repair. The tool has some problems with hold time checks and clock reconvergence pessimism removal that can lead to overly pessimistic calculations in some cases. While it is nice to know that there may be a timing problem due to crosstalk, being able to do something about it in a consistent and straightforward manner is even better. This is not the case with the current flow.

In a more ideal second generation flow, most (if not all) timing analysis should be crosstalk aware. This includes at least incremental placement (with routing estimates or actual routing information) where buffer/inverter insertion and up/downsizing can have a positive effect on crosstalk induced delays. Having a crosstalk timing in-place optimization (IPO) tool would improve the ability of the flow to fix the real crosstalk induced timing problems based on the correct information rather than indirectly trying to fix the problems through transition time constraints etc. The clock tree synthesis (and clock routing) tool should also be crosstalk aware to ensure that the proper precautions are taken (automatically) to minimize the timing effects of crosstalk on clock nets.

Finally, the crosstalk delta delay calculation needs to be implemented such that the same clock edge travelling through (at least partially) the same clock tree does not have different delays calculated for startpoint and endpoint for the common sections of the clock tree.

9.0 Acknowledgements
Rainer Mann from Synopsys Denmark provided valuable assistance and feedback.

10.0 References