# Analysis, Reduction and Avoidance of Crosstalk on VLSI Chips

Tilmann Stöhr, Markus Alt, Asmus Hetzel, Jürgen Koehl
IBM Entwicklung GmbH, Schönaicher Straße 220, 71032 Böblingen, Germany
E-Mail: tstoehr@de.ibm.com

## Abstract

*As chip size and design density increase, coupling effects (crosstalk) between signal wires become increasingly critical to on–chip timing and even functionality. A method is presented to analyze crosstalk while taking into account timing relationship and timing criticality between coupling wires. The method is based upon the geometrical layout of the wires (adjacency), the signal slopes on the wires (circuit driving capability) and timing considerations.*

*Based on these wire characteristics, a pattern driven routing tool imbeds the crosstalk critical nets in non-adjacent wiring tracks for crosstalk avoidance. The pattern driven routing capability may also be used for rerouting crosstalk critical nets of an already existing routing for crosstalk reduction.*

*The crosstalk analysis and the routing tool described in this paper were used in three generations of VLSI processor chip designs for IBM's S/390 computers, always resulting in crosstalk-resistant hardware.*

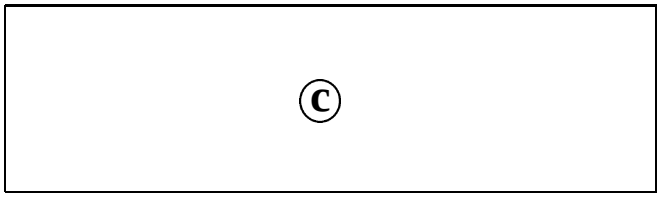
## 1 Introduction

Crosstalk is a well–known phenomenon at all levels of electronic packaging from system level cables through wires on printed circuit boards and multi–chip–modules to chip level routing. It is an electromagnetic effect due to coupling capacitances and inductances between currents in electrical conductors.

Crosstalk causes undesired signal noise to be coupled from an active line (*aggressor*) into a quiet line (*victim*). Depending on its magnitude, the induced noise onto the victim may influence the timing behaviour of the victim signal by increasing its setup time. It may even cause failure by inducing false pulses or causing false signal levels (see reference [cat]) which may be propagated through the circuit.

With increasing integration density and reduced cycle times, these effects become more visible and more destructive, so they need to be handled more carefully. Crosstalk needs to be considered in particular on VLSI chips with sub–micron structures and today's large die sizes.

A good overview of the electrical parameters determining crosstalk can be found in reference [vit].



Various transient analysis techniques to estimate noise are available. In most cases, the problem can be modeled as a linear circuit. Specialized linear model reduction techniques (see references [pil] and [fel]) help minimize model complexity and computational cost, as applied in references [she] and [dev].

A practical way to reduce computational cost and time is to separate the model extraction from the crosstalk analysis. The results of the model extraction for a small number of coupled wires with different wire geometries is repeatedly applied to the physical routing data of the VLSI chip. Both capacitive and inductive coupling are taken into account for the model extraction. A complete RLC–extraction of the chip routing is avoided.

Based on that approach, we present new methods for the analysis, reduction and avoidance of crosstalk, used successfully on VLSI logic chips developed by IBM Böblingen for three generations of S/390 processor chips. The program package described in this paper applies to IBM's most recent *"S/390 Parallel Enterprise Server Generation 4"*. The overall design methodology is presented in reference [koe]. Table 1 lists the key characteristics of the technology used.

| | |
|---|---|
| technology name | SA–12 |
| base technology | CMOS / 5 layers of metal |
| eff. channel length | $L_{eff}$ = 0.18 $\mu$m |
| min. feature size | 0.63 $\mu$m wire width / spacing |
| clock cycle | typically 4.5 ns |
| signal slopes | $<$ 0.4 ns |
| application | IBM System/390 G4 Server |

Table 1: Properties of technology used

The method for crosstalk analysis described in this paper is based on a 6–step approach. Step 1 is executed only once for each new technology, while steps 2 through 6 represent the chip dependent crosstalk analysis:

1. extract the electrical parameters for a base set of adjacent wire configurations, like 2 to 3 coupled lines with different widths and spacings
2. extract the geometry of the complete chip routing spacings
3. calculate the adjacency lengths between a chosen wire segment (victim) and its neighbors (aggressors) considering the wire widths and spacings
4. determine the circuits driving victim and aggressor lines and, from the circuits' timing rules, calculate the effective output resistances influencing the signal slopes and the amount of coupled noise
5. consider the lower acceptable noise margin for exceptionally noise–sensitive input pins (e. g. pass–gate inputs) versus the standard CMOS inputs of logic circuits

6. read the switching times of victim and aggressor lines from the chip's timing report and disregard all timing–uncritical wire adjacencies

# 2 Crosstalk on VLSI Chips

As the structures become smaller, die sizes become larger, and circuits become faster, crosstalk between two wires on a chip increases. In a simplified model, coupling between two lines can be considered a capacitive voltage divider (see figure 1). Reference [gal] describes a crosstalk estimation and analysis technique based on capacitive coupling only.

However, inductive coupling cannot be neglected and is therefore considered in the modeling in section 3. For inductances, a similar voltage divider can be modeled. The value of the mutual inductance versus the intrinsic inductance of coupled wires determines the magnitude of the induced noise.

Higher integration causes larger mutual capacitance "$C_m$" and smaller intrinsic capacitance "$C_i$" of a line, both worsening crosstalk. The same is valid for the inductive coupling as the mutual inductance "$L_m$" in particular grows significantly with denser structures.

Neglecting the output resistances "$R_{out}$" of the driving circuits leads to incorrect model behaviour. The coupled noise would then be independent of the coupling length as all $C_i$, $C_m$ , $L_i$ and $L_m$ are proportional to the wire length. Taking the output resistances $R_{out}$ and the input capacitances of the receiving circuits into account, the model in figure 1 shows a crosstalk voltage nearly proportional to the coupling wire length, as expected.

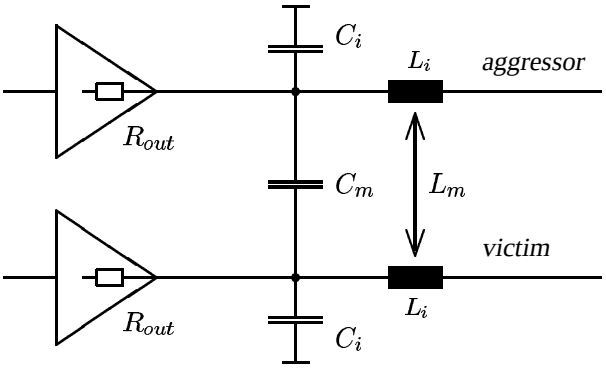



Figure 1: Simple model of coupled wires

Larger die sizes increase the average wire length on a chip, thus increasing the potential for excessive crosstalk. New technologies offer faster circuits producing faster switching output signals. This causes higher amplitudes of coupled noise across $L_m$ and $C_m$ in particular (see figure 1), again increasing the potential for excessive crosstalk.

Crosstalk may occur between both horizontally and vertically adjacent wires, i. e. wires on the same routing layer are coupling to each other as well as wires on different routing layers running on top of each other.

Wires with "diagonal" adjacency must be considered as well (see figure 2). With increasing distance between aggressor and victim in any direction, the induced coupled noise decreases drastically.

The wire widths heavily influence the coupling, as all components of the line model in figure 1 ($L_m$, $L_i$, $C_m$ and, in particular, $C_i$) depend on the conductor width.

horizontal adjacency: A1–V A5–V
vertical adjacency: A3–V
diagonal adjacency: A2–V A4–V

A1 ... A5 = aggressors
V = victim

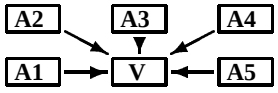



Figure 2: Chip cross section with adjacent wires

# 3 Modeling of Crosstalk

In addition to the geometrical arrangement of the wires, other important factors influence the crosstalk noise coupled into the victim net. Two of these are the driving capability of the circuits on the nets and the timing relationship between aggressor and victim. The following subsections discuss these factors in detail.

## 3.1 Extraction of Wire Parameters

A sufficiently exact crosstalk analysis of a given chip routing to predict potential crosstalk problems is strongly dependent on the availability of an accurate model for coupled lines and of analytical methods for capacitance and inductance calculation. Detailed analyses of capacitance modeling on chip level are found in references [sak] and [del], the latter also including inductances. As mentioned in the introduction (section 1), full chip level extractions of electrical parameters are not practical due to data volume. In our approach, only a small number of adjacency configurations are being investigated in a first step.

The noise voltage induced into a single victim wire is calculated with a detailed wire model. *AS/X*, an internal IBM circuit simulation tool similar to SPICE (see reference [spc]) is used for the analysis. It simulates both the circuits driving victim and aggressor lines as well as the coupled lines based on their R–, C– and L–components. A line is broken down into several hundred pieces of discrete RLC–components representing the wire behaviour.

A 3D extraction tool is used to determine the lines' C– and L–components. "3D" in this context means considering not only adjacent lines on the same layer of the chip, but also crossing lines on the layers above and below as well as the wider power lines on all layers. The power lines as current return paths have a great influence onto the self–inductance of chip wires and therefore onto the inductive coupling.

For the *SA–12* CMOS technology described in table 1, an induced peak noise of $V_{DD}/3$ is allowed. This leaves sufficient margin for additional power noise. Detailed *AS/X* circuit simulations show that noise above this peak value at a circuit's input pin causes the circuit to propagate the noise pulse. Avoiding noise propagation ensures the correct functioning of the circuits in a logic path.

The acceptable height of a noise pulse at a circuit's input pin is dependent on its width (duration). The DC noise limit is used as the worst case lower limit for noise of any duration. Thus, noise pulse width may be neglected without missing any critical coupling.

## 3.2 Geometry

The approach in this paper is to determine the coupling between a pair of horizontally directly adjacent wires (minimum spacing, minimum width) in a first step. Other geometries (different spacing and width, vertical and diagonal neighbourhood) are then calculated relative to that reference. A critical maximum length "$L_{CRIT}$" for direct adjacency is determined such that the receiving circuit on the victim wire will not switch due to a noise peak.

Using *AS/X* to simulate the typical–case scenario consisting of identical driving circuits on aggressor and victim line, $L_{CRIT,REF}$ = 8 mm is found for the *SA–12* technology.

For other wire geometries, relative coupling factors "$F_{wire}$" are introduced which describe the induced peak noise compared to direct horizontal adjacency for a given coupled length. Direct horizontal adjacency causes the highest coupling of all possible geometries ($F_{wire} = 1.0$). Therefore, all relative coupling factors are $F_{wire} \leq 1.0$.

| distance H: [*channels*] V: [*planes*] | width victim [*channels*] | width aggressor [*channels*] | relative coupling factor $F_{wire}$ |
|---|---|---|---|
| 1 H | 1 | 1 | 1.00 |
| 1 H | 2 | 2 | 0.89 |
| 2 H | 1 | 1 | 0.69 |
| 2 H | 2 | 2 | 0.63 |
| 3 H | 1 | 1 | 0.52 |
| 3 H | 2 | 2 | 0.47 |
| 2 V | 1 | 1 | 0.56 |
| 2 V | 2 | 2 | 0.68 |

Note: H = horizontal adjacency
V = vertical adjacency

Table 2: Relative coupling factors (wire)

The relative coupling factors of different wire geometries are shown in table 2. This table lists only some example values from the complete list of all wire geometries used by a given routing tool and of all possible vertical and diagonal adjacencies. For wire distances of $\geq 4$ channels, the relative coupling factors are so small that crosstalk can be neglected for all practical purposes.

The relative coupling factor $F_{wire}$ for a given geometry means that the induced noise is smaller by the factor $F_{wire}$ compared to the same length of directly adjacent wires, or that the acceptable adjacency length for that geometry is longer by $1 / F_{wire}$.

A victim may have, of course, more than one aggressor coupling into it. Without looking into the timing of the signals, all aggressors must be considered switching simultaneously in a worst case scenario. As the induced noise depends on the product of the geometrical adjacent length and the geometry weight factor $F_{wire}$ from table 2, these so–called "weighted adjacent lengths" can simply be summed to get the "total weighted adjacent length" coupling into a victim.

If the total weighted adjacent length exceeds the above defined value $L_{CRIT,REF}$, the victim net is considered "critical". Excessive crosstalk is very likely to occur on this net. Critical nets are handled by additional measures described in sections 5 and 6.

## 3.3 Circuit Driving Capability

The driving capabilities of the circuits on the aggressor and victim nets are represented by their output resistances. A smaller output resistance causes a steeper slope of a switching signal at this pin. An aggressor net driven by a circuit with small output resistance has steep signal slopes, and more noise will be coupled into an adjacent victim wire. A victim driven by a circuit with small output resistance, however, will be less sensitive for crosstalk, as the low "damping" resistor at the wire input will allow less noise to be induced via the mutual capacitance and inductance between aggressor and victim.

*AS/X* simulations have been run with circuits of different driving capability on the same pair of coupled lines. The circuit output devices have been varied from very small to very large transistor sizes on both victim and aggressor, resulting in corresponding effective output resistances from $< 100\,\Omega$ to $> 1000\,\Omega$. Table 3 shows the coupled noise relative to the "nominal" case with identical aggressor and victim driving circuits.

As the geometrical arrangement of the coupled lines is expressed in relative coupling factors, the circuit output resistance is taken into account in a similar manner. The (geometrically) weighted adjacent lengths are weighted additionally with the relative coupled peak noise factor "$F_{circuit}$" as listed in table 3.

| aggressor output resistance [Ω] | victim output resistance [Ω] | relative peak noise from AS/X simulations | relative peak noise according to equation (1) |
|---|---|---|---|
| 1095 | 1095 | 0.93 | 1.00 |
| 452 | 452 | 1.00 | 1.00 |
| 147 | 147 | 1.05 | 1.00 |
| 1095 | 147 | 0.35 | 0.34 |
| 452 | 147 | 0.62 | 0.61 |
| 147 | 452 | 1.61 | 1.61 |
| 147 | 1095 | 2.82 | 2.90 |

Table 3: Relative coupling factors (circuit)

For simple modeling of the influence of the output resistances, a simple mathematical equation is chosen, which efficiently and very closely approximates the discrete values in table 3. *AS/X* results serve as base to derive an equation on a "best fit" basis.

Equation (1) approximates the exact value $F_{circuit}$ from the above mentioned *AS/X* simulations within 10% for the *SA–12* technology.

$$F_{circuit} = \frac{R_{out,victim} + 380\,\Omega}{R_{out,aggressor} + 380\,\Omega} \qquad (1)$$

This approximation equation is found under the condition $F_{circuit}$ = 1.0 for identical circuits on both coupled wires with $R_{out,victim} = R_{out,aggressor}$. For identical victim and aggressor circuits, a maximum allowed adjacent length $L_{CRIT,REF}$ = 8 mm is found with *AS/X* simulations, as mentioned in section 3.2.

For different wire geometries (width and spacing) and different driving circuits (output resistance) the weighted adjacent length "$L_{WGHT}$" is defined. It is calculated from the geometrical adjacent length "$L_{ADJ}$" (victim and aggressor running in parallel) according to equation (2).

$$L_{WGHT} = L_{ADJ} \cdot F_{wire} \cdot F_{circuit} \qquad (2)$$

Hence, the total crosstalk induced into a victim without considering timing is the sum of the contributions of all aggressors. Under the assumption of a peak noise of $V_{DD} / 3$ for 8 mm coupled length of directly adjacent wires, the total noise voltage "$V_{tot}$" of all aggressor segments "$i$" is calculated according to equation (3).

$$V_{tot} = \frac{V_{DD}}{3} \cdot \sum_i \left( \frac{L_{ADJ,i}}{8\,mm} \cdot F_{wire,i} \cdot F_{circuit,i} \right) \qquad (3)$$

Equation (3) implies a peak noise of less than $V_{DD}/3$ if the following condition is satisfied:

$$\sum_i (L_{ADJ,i} \cdot F_{wire,i} \cdot F_{circuit,i}) \quad < \quad 8\,mm \qquad (4)$$

The terms in equations (3) and (4) may all be associated with different aggressor nets. Some of them, however, may result from different segments belonging to the same aggressor net. All these adjacent segments are added up in the same way according to equation (4).

### 3.3.1 Comparison to Analytical Model

The dependency of the coupled peak noise from the circuit output resistances and the coupled wire length is derived in another way in reference [vit] based on a crosstalk model as shown in figure 3, using an analytical noise model.

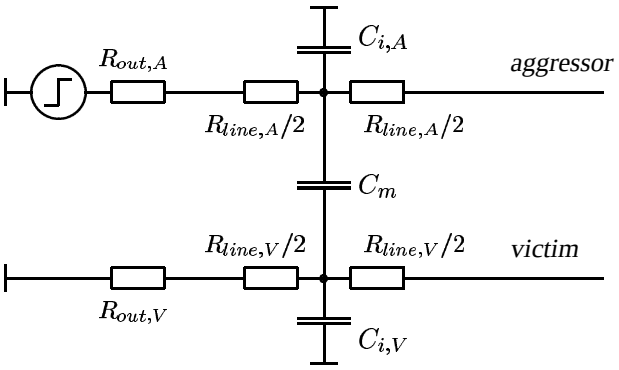



Figure 3: Capacitive model (T–circuit) of coupled wires

The aggressor line is driven by a voltage source (stepping $0 \rightarrow V_{DD}$) with an intrinsic resistance of "$R_{out,A}$", the victim line is connected to ground via a quiet circuit represented by just its intrinsic resistance "$R_{out,V}$".

The two coupled lines are modeled by the discrete values of their intrinsic capacitances "$C_{i,A}$" and "$C_{i,V}$" and the mutual capacitance "$C_m$" between the two lines.

To enhance the model described in reference [vit], the longitudinal resistances "$R_{line,V}$" and "$R_{line,A}$" of the lines are added. They are modeled by the resistors on either side of the discrete capacitors with half the longitudinal resistance of the lines each.

According to reference [vit], the peak noise voltage of the model in figure 3 is

$$V_{noise} = \frac{V_{DD}}{\frac{R_{out,A} \cdot C_{i,A}}{(R_{out,V} + R_{line,V}/2) \cdot C_m} + \frac{C_{i,V}}{C_m}} \qquad (5)$$

Equation (5) was run through the mathematical tool "Maple" to verify the values in table 3. The simplified model in figure 3 together with the step generator, as opposed to the behaviour of a "real" driving circuit, is somewhat pessimistic. Indeed, the peak noise voltages derived from this analytical model are always between 10% and 20% higher than in *AS/X* simulations. This is a satisfying confirmation of the coupled line model chosen in this paper.

## 3.4 Noise Sensitivity of Input Pins

In some cases, the threshold voltage of certain circuit input pins may be lower than for the typical CMOS library elements. Pins with lower threshold voltages may be used to decrease switching times. Either the transistors are designed to have lower gate threshold voltages or pass gate inputs may be used, e. g. for latches. These lower noise margins imply a higher sensitivity to crosstalk.

The crosstalk analysis tool in this paper easily allows for these lower noise margins by another correction factor to the maximum allowed adjacency length. Like the weight factors for different geometries (see section 3.2) and for different circuit driving capability (see section 3.3), another weight factor is introduced for nets connected to noise sensitive pins.

The acceptable physical adjacency length of a victim wire decreases proportionally to the noise margin of the most sensitive input pin connected. The noise margin of a standard CMOS input pin allowing sufficiently for additional power noise is given as $V_{margin,ref} = V_{DD}/3$.

Victim nets having pins with lower noise margin $V_{margin}$ are weighted with a factor $F_{noise} > 1$. The weighted adjacency length increases with $F_{noise}$ allowing less physical adjacency length.

$$F_{noise} = \frac{V_{margin,ref}}{V_{margin}} = \frac{V_{DD}/3}{V_{margin}} \qquad (6)$$

The total weighted adjacency length of a victim defined in equation (4) is multiplied with $F_{noise}$. This value must be less than the reference critical length ($L_{CRIT,REF}$ = 8 mm for *SA–12*).

$$F_{noise} \cdot \sum_i (L_{ADJ,i} \cdot F_{wire,i} \cdot F_{circuit,i}) < 8\,mm \qquad (7)$$

for all "i" aggressor segments.

## 3.5 Timing Behaviour

Many of the nets identified as "crosstalk critical" when considering geometry (see subsection 3.2) and circuit output resistance only (see subsection 3.3) can further be eliminated by considering their timing. Most of our CMOS VLSI chips analyzed for crosstalk are synchronous designs, i. e. they have paths starting at a latch output, passing through combinatorial logic only and ending at another latch's input. A signal is launched by a clock pulse, propagated through its path and is caught at the receiving latch by another, synchronous clock pulse. On some chips, however, different asynchronous clock domains are implemented.

A victim net may be discarded from the list of critical nets if the crosstalk pulse induced occurs early enough before the signal's latest allowable switching time. To be able to specify "early enough", the following descriptors of the timing behaviour are defined:

| | |
|---|---|
| $AT$ | **arrival time:** time when a logic signal reaches its final state in a clock cycle |
| $RAT$ | **required arrival time:** time when a logic signal must have reached its final state before getting latched |
| $Slack$ | **slack time:** time "reserve" between arrival time and required arrival time ($RAT - AT$) |
| $T_{CLK}$ | **clock pulse:** time when the logic signals are latched at the receiving latches |
| $T_{margin}$ | **safety margin:** time to allow induced noise to fade away ($\simeq$ signal slope time) |

A victim net with a positive slack value "$Slack$" has this extra time period until it is latched after propagation through its path. Hence, a

noise pulse induced within that time period or earlier is not harmful. Figure 4 shows the relationship of the above defined timing values in a timing diagram.

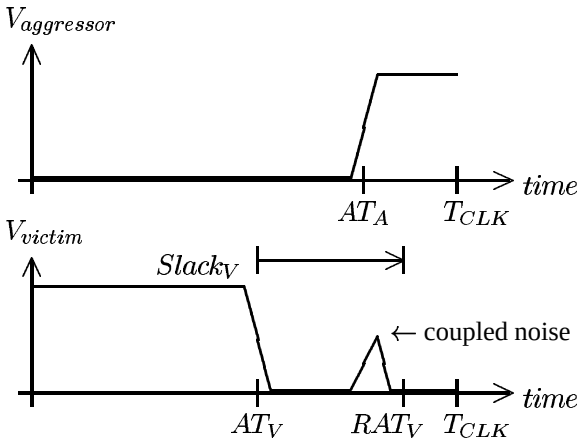



Figure 4: Timing descriptors for coupled nets

As the induced noise pulse on the victim net needs some time to fade away, a time margin $T_{margin}$ is introduced. From experience, $T_{margin}$ is comparable to the duration of the signal slope on the aggressor line. Therefore, $T_{margin} = 0.4\,ns$ has been chosen for the *SA–12* technology.

In summary, equation (8) describes the condition for a victim's stable state not to be influenced by a coupling aggressor. All nets which fulfill this relation may be discarded as critical crosstalk problems. This optional function is called the "timing filter", applied to each pair of aggressor and victim lines.

$$AT_A \quad < \quad AT_V + Slack_V - T_{margin} \tag{8}$$

There are two situations in which equation (8) may not be applied to discard critical nets. First, clock nets usually switching at exactly the critical latching time are always considered critical aggressors and critical victims (a noisy clock pulse causes functional problems). Second, the induced noise may occur unpredictably at any point in time between coupled nets belonging to different clock domains (asynchronous nets).

# 4 Analysis of Crosstalk

Based on the model presented in section 3, a "C" program called *CrossTalk* implements the complete presented method. The program reads a parameter file containing the relative (geometrical) coupling factors $F_{wire}$ as defined in section 3.2 and the critical length $L_{CRIT,REF}$ as given in section 3.3.

The chip data, in particular all net segments and the driving circuits of all nets, is read from the design data–sets. These are a number of binary files corresponding to the in–core data structures of the logical and physical chip data. Further, a "circuit list" containing the output resistances of all circuits is read. The timing data (slacks, arrival times, clock cycle times) of the design is read from the "net file" generated in a preceding timing analysis run.

*CrossTalk* extracts all adjacent segments as given in the $F_{wire}$ parameter table (e. g. up to a distance of 3 channels). It then finds the driving circuits for the coupling nets and calculates the corresponding relative coupling factor $F_{circuit}$ according to equation (1). With these data, the weighted adjacent lengths according to equation (2) are calculated and added up. All victim nets with a total weighted adjacent length below $L_{CRIT,REF}$ are discarded then from the list of critical nets.

In a final step, the "timing filter" discards the timing uncritical nets which satisfy equation (8). As already mentioned, this "timing filter" does not discard any clock nets from the report, nor does it discard adjacencies between nets belonging to different clock domains.

In the *SA–12* technology of table 1, a fully populated 12.8 mm chip has been analyzed using *CrossTalk*. The design contains 342,000 nets with 3,490,000 wire segments. To show the advantages of considering geometric, electrical and timing properties of the nets, the results of three scenarios of *CrossTalk* are listed in table 4.

| scenario (active option, cumulative) | critical nets found | CPU time $[mm:ss]$ | memory used $[MB]$ |
|---|---|---|---|
| 1) geometry only | 246 | 9:56 | 623 |
| 2) circuit $R_{out}$ | 97 | 10:40 | 679 |
| 3) timing of nets | 46 | 14:41 | 683 |

Table 4: Comparison of different program scenarios

Introducing the dependency on the output resistance of the circuits in scenario 2) of table 4, and activating the timing filter in scenario 3), reduces the number of incorrectly identified critical adjacencies significantly. This minimizes the effort to eliminate the critical nets reported. Strategies to solve the remaining problems are discussed in section 5 and section 6.

*CrossTalk* reports the nets found as critical in a comprehensive report file which shows all characteristics of the coupling nets, i. e. whether a net is victim or aggressor, the weighted adjacent length of the coupling segments, its clock domain (cycle time), its timing descriptors (see section 3.5) and its driving circuit's output resistance. Here's a small example of a report file:

```
CrossTalk report for project PUCHIP
generated:  Wed Feb 04 13:26:20 1998
Timing filter option:  ON
```

| weighted adj | len | type | cycle | slack | AT | net name | Rout |
|---|---|---|---|---|---|---|---|
| victim: | | CLK | | | | ACLK | 452.6 |
| total: | 475 | | | | | | |
| aggressor: | 138 | CLK | | | | BCLK | 452.6 |
| aggressor: | 100 | NET | 6.500 | 0.540 | 4.501 | ERRORCOM1 | 276.4 |
| aggressor: | 84 | CLK | | | | ADELAYCLK | 452.6 |
| aggressor: | 52 | NET | 6.500 | 0.424 | 5.789 | TOADDER3 | 153.9 |
| aggressor: | 51 | NET | 6.500 | 0.566 | 4.651 | ERRORCOM2 | 276.4 |
| aggressor: | 50 | NET | 6.500 | 0.980 | 4.002 | ERRORCOM3 | 276.4 |
| | | | | | | | |
| victim: | | NET | 6.500 | 0.234 | 6.023 | NET0003 | 264.4 |
| total: | 474 | | | | | | |
| aggressor: | 138 | NET | 6.500 | 1.201 | 5.202 | NET0004 | 993.9 |
| aggressor: | 100 | CLK | | | | CCLK-LATE | 452.6 |
| aggressor: | 100 | CLK | | | | CCLK1LATE | 452.6 |
| aggressor: | 84 | NET | 6.500 | 0.766 | 5.700 | FROMADD3 | 145.6 |
| aggressor: | 52 | CLK | | | | BCLK-LATE | 452.6 |

# 5 Reduction of Crosstalk

The coupling nets remaining as "critical" after applying the methods described in sections 3 and 4 must be modified to reduce their criticality without worsening crosstalk between other nets.

A straight–forward net length minimizer proved to work fairly well for the chips based on the *SA–12* CMOS technology with up to 12.8 mm die size and more than 340,000 nets. A net optimizing program reroutes the remaining critical nets reported by *CrossTalk* trying to minimize the net length and the number of vias used. This results in straightening out those nets while the local routing tool

originally tried to minimize the use of routing resources, inserting many "edges" and "bends" into a wire.

This procedure requires little CPU time and reduces the number of crosstalk critical nets considerably. A subsequent crosstalk analysis proves that only a small number of these nets remain critical on all chips mentioned above. These remaining nets can be checked manually, using *AS/X* to exclude all worst case assumptions of the model used in this paper or they can be manually rerouted, e. g. by moving adjacent nets apart for a fraction of their adjacent length.

Rerouting crosstalk critical nets using this basic net optimization method is admittedly not deterministic but has proven to work well in practice, in particular on the last two generations of VLSI processor chips designed in technologies earlier than *SA–12*.

A deterministic reroute method is applied to the high density *SA–12* chips of the latest generation. It uses "pattern driven routing" as described in section 6 to reroute all nets considered critical by a preceding crosstalk analysis of the completely routed chip.

The victim nets found to be critical are collected in a class *Victims*, the aggressor nets in a class *Aggressors*. Both classes are associated with special wire codes *wire_aggressor* and *wire_victim*, respectively. These wire codes are assigned to routing patterns ("shape classes") disallowing a *wire_aggressor* to be routed adjacently to a *wire_victim* and vice versa.

The "pattern driven routing" tool now attempts to reroute all victims and aggressors according to the given routing patterns. If one or more segments of a wire fail rerouting, the original routing will be kept to avoid generating incomplete nets.

On a very dense chip however, this may cause some crosstalk problems to remain unsolved by this automated reroute procedure. Additional effort by the designer will then be necessary. A proven method of manual rerouting is "jogging" of a long victim net between or around its aggressor wires to reduce the influence of one single aggressor. An algorithmic solution of this manual approach is presented in reference [jhj].

Figure 5 shows an overview of the proposed design flow for crosstalk reduction of an existing chip routing.

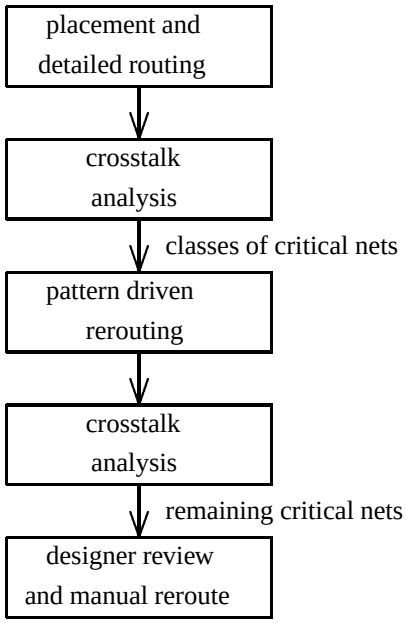



Figure 5: Process Flow for Crosstalk Reduction

Running the above "pattern driven routing" method on the already mentioned 342,000 nets VLSI processor chip, major crosstalk improvements can be achieved as listed in table 5.

A reduction of the number of victims to zero by using automated rerouting is not possible on this very dense chip. In any case, the manual effort to improve or remove the remaining critical adjacencies is minimized, as the maximum weighted adjacency length for the worst victim net is also reduced.

| rerouting applied to | victim nets found | aggressor nets found | maximum total adj. length [$\mu$m] |
|---|---|---|---|
| 1) none | 46 | 95 | 10309 |
| 2) victims only | 32 | 57 | 10145 |
| 3) victims + aggressors | 21 | 34 | 6902 |

Table 5: Results of critical nets' rerouting

Further effort is being invested into the "pattern driven routing" tool which capable of routing any number of wires grouped into classes following certain constraints. This new method generates an initial chip routing attempting to avoid crosstalk altogether.

# 6 Avoidance of Crosstalk

For the increasing coupling effects in future technologies, the simple reroute strategy described in section 5 may no longer be sufficient. Instead, a method is needed that avoids crosstalk problems while routing. In contrast to more general approaches interacting with all aspects of the chip physical design (as discussed in reference [kir]) the method in this paper simply applies a detailed routing tool controlled by crosstalk constraints.

Prior papers like reference [gao] dealt with the improvements of channel routing algorithms capable of generating routing solutions that satisfy the relative positions of the chip wires. In contrary to our proposal, even short crosstalk uncritical wires were considered and affected, resulting in a reduction of the routing capability within a given area.

As crosstalk is strictly a local phenomenon it is handled within detailed routing (local routing) rather than in global routing. The final arrangement of all wire segments is determined within detailed routing, where the crosstalk relevant parameters (see section 3) can be extracted. Moreover, the detailed router usually has sufficient freedom for the assignment of wire segments to channels to avoid the most critical coupling configurations. However, detailed routing is typically one of the most CPU time and memory intensive tasks in physical chip design. Therefore, the detailed router is guided by simple geometrical restrictions for crosstalk avoidance rather than by a complete complex electrical wire model.

*XRouter* is a program package designed for high–performance VLSI chip routing developed at the Research Institute for Discrete Mathematics of the University of Bonn, Germany. It uses a two stage global / detailed routing scheme. Global routing restricts the area where a net may be routed to a small part of the chip based on Steiner tree constructions and congestion estimates. Detailed routing is based on a sequential rip–up and reroute approach.

*XRouter* is capable of performing detailed "pattern routing". The underlying principle assumes that any piece of metal ("shape") placed on the wiring grid is associated with a certain class ("shape class"). Using input parameters, up to 32 shape classes may be defined. Each shape class contains predefined shape types (e. g. blockages, pin areas, power rails) associated with nets or net parts of certain groups.

"Shape classes" define patterns to be avoided during the final routing. For crosstalk purposes, two shape classes $SC_{couple}$ and $SC_{quiet}$

may be defined. Crosstalk sensitive wires are associated with $SC_{couple}$, non–sensitive wires with $SC_{quiet}$. Then the pattern

$$(SC_{couple}, SC_{couple})$$

prevents the router from putting crosstalk sensitive wires into adjacent channels, while the pattern

$$(SC_{couple}, \text{FREE}, SC_{couple})$$

prevents the router from putting crosstalk sensitive wires on both sides of an unused channel. Two sensitive wires on both sides of another segment associated with $SC_{quiet}$, however, are permitted due to additional shielding.

Arbitrary forbidden patterns may be specified individually for all planes and separately for horizontal and vertical adjacencies. Additionally, patterns can be restricted to apply only to the global part of the nets to improve pin accessibility.

*XRouter* is designed to handle a large number of different patterns simultaneously. It uses a multi–level pattern evaluation scheme involving several look–up and caching steps. Therefore, performance is mostly affected by the size of the patterns, i. e. the number of channels they involve. The solvability of the routing task (e. g. contradicting patterns given by the user) significantly influences the performance, while the number of patterns given does not limit solvability and performance considerably.

The following general approach will be used for crosstalk avoidance with pattern routing in future chip physical designs:

- Determine potentially critical victim and aggressor nets by analysis of circuit driving capability, wire width and estimated routing length, e. g. by Steiner tree approximations from the preceding placement step or the global router.
- Divide the critical nets into "orthogonal" sets whenever possible, i. e. group these nets so that crosstalk is critical only between members of the same group. This is useful in particular for multistage–clock designs where nets are known to be "quiet" in certain time intervals.
- Code shape classes to disallow nets of the same group to run in parallel without sufficient distance or intermediate shields.
- Run the detailed router under the terms of these additional shape classes.

Not all potential crosstalk problems on any given chip can be treated this way, but for the majority of nets the allowed neighbourhood is controllable by patterns to sufficiently avoid coupling. More details about this "pattern driven router" and its usage are found in references [kru] and [het].

Figure 6 shows an overview of the proposed design flow for crosstalk avoidance during initial routing of a chip.

The "shape classes of long nets" are generated by the netlength estimate of the global router in figure 6. They contain all potential aggressor nets in $SC_{aggressor}$ (global connections longer than a critical value, e. g. 2 mm for *SA–12*) and all potential victim nets in $SC_{victim}$ (global connections longer than a critical value, e. g. 5 mm for *SA–12*). The following routing patterns are defined to prevent adjacent routing of victim to victim, victim to aggressor and aggressor to victim:

$$(SC_{victim}, SC_{victim})$$

$$(SC_{victim}, SC_{aggressor})$$

$$(SC_{aggressor}, SC_{victim})$$

Controlling the "pattern driven router" in the above way using 2 shape classes and 3 patterns will generate a routing optimized for little crosstalk. On a very dense chip, however, not too many nets should be included into the 2 shape classes. Local wiring congestion will cause the router to generate long detours for some of the specified nets. Chip timing will be affected, even though crosstalk problems will be avoided. Additionally, the run time of the router will increase unacceptably up to a magnitude in CPU time.

Good percentages for the mentioned 342,000 nets VLSI processor chip are:

- less than 5% of all nets in $SC_{aggressor}$ (medium nets)
- less than 1% of all nets in $SC_{victim}$ (long nets)

To achieve the maximum automated crosstalk minimization, the processes of *Crosstalk Avoidance* and *Crosstalk Reduction* are combined. The result of the crosstalk analysis in figure 6 is fed into the "pattern driven rerouting" of figure 5. This reduces the number of eventually remaining crosstalk critical adjacencies after the initial "pattern driven detailed routing" in figure 6.

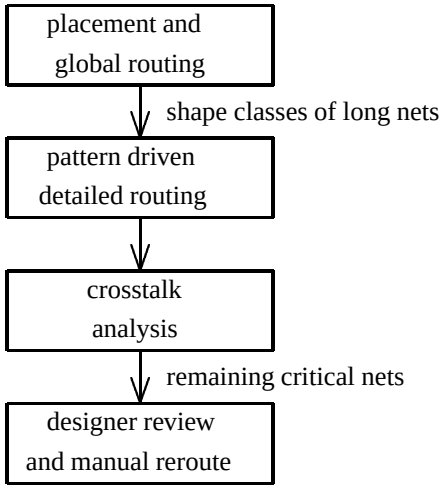



Figure 6: Process Flow for Crosstalk Avoidance

The methods discussed in this section have been applied to the previously mentioned 342,000 nets VLSI processor chip. Major crosstalk improvements regarding the number of critical nets are achieved as listed in table 6. To design a minimum crosstalk VLSI chip, the combination of both crosstalk avoidance routing and crosstalk reduction rerouting is recommended.

| crosstalk process applied | victim nets found | aggressor nets found | maximum total adj. length [$\mu$m] |
|---|---|---|---|
| 1) none | 46 | 95 | 10309 |
| 2) rerouting only | 21 | 34 | 6902 |
| 3) avoidance only | 14 | 35 | 7567 |
| 4) avoidance + rerouting | 5 | 7 | 6309 |
| 5) avoidance + rerouting | 1 | 1 | 6089 |

Table 6: Results of crosstalk processes

Run 5) in table 6 shows that it is possible to generate a virtually crosstalk–free chip design using the described process of crosstalk avoidance with a subsequent crosstalk reduction. However, this run needed 5 times more CPU time for the initial pattern driven routing (82 h versus 18 h). Additionally, the router was forced to use wiring detours due to an excessive number of nets in the non–adjacent

shape classes, resulting in a netlength increase from 194.1 meters to 195.7 meters.

These wiring detours causing additional RC–delays impact the timing of some logic paths. The worst slack of all paths went up by 120 ps at a cycle time of 4.5 ns. Therefore, timing always needs to be considered when running the presented crosstalk process. Maximum automated crosstalk avoidance with minimum manual effort to remove the remaining critical adjacencies can be traded for marginally slower chip timing.

However, runs 1) through 4) in table 6 resulted in identical chip timing and similar wiring lengths.

The small amount of remaining critical adjacencies has to be reviewed by the chip designer. Subsequent detailed *AS/X* circuit simulations of these nets show if the presented method was too pessimistic due to some worst case assumptions, or if the adjacencies need additional treatment. Again, "jogging" of the long victim nets between or around its aggressor wires reduces the total coupled noise. Running the automated crosstalk avoidance process minimizes the manual effort to fix the few remaining critical adjacencies.

# 7 Summary and Conclusions

A new method is presented which comprehensively identifies and corrects on–chip crosstalk on large VLSI designs. The extraction of coupling parameters determined by wire geometry, circuit driving capability, and timing behaviour is separated in this procedure from working with the chip specific physical routing data. The implementation of this methodology in a "C" program runs quickly and efficiently.

Taking the mentioned properties of aggressor and victim nets into account, a sufficiently exact analysis of the crosstalk criticality of all nets is achieved. No unnecessary worst case assumptions must be applied as safety margin for side effects not considered in earlier proposals. Crosstalk analysis can be performed easily, quickly and with reliable results.

As our *CrossTalk* program runs on flat chip designs, coupling from and to circuits' internal wires will not be handled, as these are not visible in the chip level design data. In particular very large circuits, such as array macros, may send or receive coupled noise to or from chip level wires, respectively. Further enhancements to the analysis program to cover this kind of crosstalk are under investigation.

The use of a "pattern driven router" in our chip designs either permits the reduction of on–chip crosstalk in an existing routing or the avoidance of crosstalk in the initial chip routing already.

Rerouting ("crosstalk reduction") is driven by shape classes (forbidden patterns) applied to nets found by a crosstalk analysis of the completely routed chip. To guide the pattern driven router from the beginning on a new chip design ("crosstalk avoidance"), the necessary net constraints are generated from the global chip routing as shape classes for long connections.

All chips treated according to this paper are free of any crosstalk related problems in the hardware, in contrast to the previous chip generation. One of these chips had a functional problem, discovered as late as during hardware bring–up, as no crosstalk anlysis was available during the design phase. The program *CrossTalk* was run on this particular chip to verify the coupling problem, and it predicted the critical adjacency exactly as observed in hardware. Running *CrossTalk* and reducing or avoiding critical adjacencies is essential to get functional hardware.

# Acknowledgements

We would like to thank the department members of the "VLSI Design Center" of the IBM Development Laboratory in Böblingen for their hints and support for the wording and the contents of this paper. In particular Günther Hutzl was very helpful about using LaTeX, and Erich Klink of the "VLSI Packaging and Physical Design" department layed the basics to initially consider crosstalk an important aspect of advanced chip physical design.

# References

[cat] I. Catt, "Crosstalk (Noise) in Digital Systems", IEEE Transactions, Electronic Computers, vol. 16, no. 6, 1967, pp. 743–763

[del] N. Delorme, M. Belleville, J. Chilo, "Inductance and Capacitance Formulas for VLSI Interconnects", Electronic Letters, vol. 32, no. 11, May 1996

[dev] A. Devgan, IBM Research, Austin, Texas, "Efficient Coupled Noise Estimation for On–Chip Interconnects", Proceedings, IEEE/ACM International Conference on Computer Aided Design, 1997, pp. 147–151

[fel] P. Feldmann, R. W. Fruend, "Reduced–Order Modeling of Large Linear Subcircuits via a Block Lanczos Algorithm", Proceedings, ACM/IEEE Design Automation Conference, 1995, pp. 474–479

[gal] L. Gal, Motorola, Austin, Texas, "On–Chip Cross Talk — the New Signal Integrity Challenge", Proceedings, IEEE Custom Integrated Circuits Conference, 1995, pp. 251–254

[gao] T. Gao, C. L. Liu, "Minimum Crosstalk Channel Routing", IEEE Transactions, Integrated Circuits and Systems, vol. 15, no. 5, May 1996, pp. 465–474

[het] A. Hetzel, Research Institute for Discrete Mathematics, University of Bonn, Germany, "Routing in VLSI Design: Special Partial Problems and a Sequential Solving Algorithm", Doctoral Degree's Thesis, 1995

[jhj] K. S. Jhang, S. Ha, C. S. Jhon, "A Crosstalk Optimizer for Gridded Channel Routing", IEEE Transactions, Computer Aided Design, vol. 15, no. 4, April 1996, pp. 424–429

[kir] D. Kirkpatrick, A. Sangiovanni–Vincentelli, "Techniques for Crosstalk Avoidance in the Design of High–Performance Digital Systems", Proceedings, IEEE International Conference on Computer Aided Design 1994, pp. 616–619

[koe] J. Koehl, U. Baur, T. Ludwig, T. Pflüger, "A Flat, Timing–Driven Design System for a High–Performance CMOS Processor Chipset", to appear in Proceedings, Design Automation and Test in Europe Conference, 1998

[kru] A. Hetzel, F. Kruse, Research Institute for Discrete Mathematics, University of Bonn, Germany "XRouter Version 05.00", Program Description and User's Guide

[pil] L. T. Pillage, R. A. Rohrer, "Asymptotic Waveform Evaluation for Timing Analysis", IEEE Transactions, Computer Aided Design, vol. 9, no. 4, April 1990, pp. 352–366

[sak] T. Sakurai, K. Tamaru, "Simple Formulas for 2– and 3–D Capacitances", IEEE Transactions, Electronic Devices, vol. ED–30, no. 2, Feb. 1983, pp. 183–185

[she] K. Shephard, V. Narayan, "Noise in Submicron Digital Design", Proceedings, IEEE International Conference on Computer Aided Design, 1996

[spc] L. W. Nagel, "SPICE2, A Computer Program to Simulate Semiconductor Circuits", Technical Report ERL–M520, University of California, Berkeley, May 1975

[vit] A. Vittal, M. Marek–Sadowska, University of California, Santa Barbara, "Reducing Coupled Noise During Routing", Proceedings, Fifth ACM SIGDA Physical Design Workshop 1996, pp. 27–33