ECCTD'01 - European Conference on Circuit Theory and Design, August 28-31, 2001, Espoo, Finland

A Framework for Automatic Analysis of Geometrically Proximate Nets in VLSI Layout

Sandeep Koranne* and Om Prakash Gangwal*

Abstract — **We address the problem of automatic analysis of geometrically proximate nets in VLSI layout by presenting a framework (named FASCL) which supports pairwise analysis of nets based on a geometric kernel. The user can specify *functions* which use parameters reported by the geometric kernel (e.g. coupling length, maximum layer). The user can also attach these functions to *conditions* and FASCL will automatically apply the function to all pairs of nets which satisfy a condition. Our method runs with sub-quadratic time complexity, $O(N^{1+k})$, where N is the number of nets and $k < 1$. We have successfully used the program to analyze circuits for bridging faults. The problem finds practical use in many other CAD applications like coupling capacitance extraction, crosstalk analysis, signal integrity analysis and delay fault testing.**

1 Introduction

With the rapid increase in design complexity and the simultaneous decrease in feature size, interconnect analysis has become a major task in computer aided verification of VLSI circuits. Coupling effects between adjacent interconnect lines can cause significant performance and signal integrity problems [1], [5]. The need of an efficient estimator of the impact of coupling effects on a given net is very important in emerging Deep Sub-Micron (DSM) and Very Deep Sub-Micron (VDSM) technologies. All reported formulations for interconnect analysis use some coupling analysis function on nets to form a sort of crosstalk risk graph [6] or state the need for analyzing *neighbouring* pairs of nets with detailed techniques [2].

Not all pairs of nets need be analyzed to the same level of accuracy and because the parameters for analysis of nets change constantly, a fast and configurable geometric kernel needs to be developed which can *apply* specific analysis techniques to a set of nets which are close to each other (we refer to such a pair as geometrically proximate). This paper describes the design and implementation of such a geometric analysis tool FASCL which is an acronym for FASt calculation of Coupling functions in VLSI Layout. Given a layout, a set of coupling functions and a set of rules, FASCL will automatically call the appropriate coupling analysis function on the set of nets which satisfy a given rule. Also, the complete analysis of a design can be done in $O(N^{1+k})$ time, where N is the number of nets (segments) in the design and $k < 1$.

A common feature in all coupling analysis formulations is the presence of an upper bound on the distance over which coupling effects are significant. In other words, all nets which are further away than this technological dependent parameter ϵ, need not be analyzed pairwise. This fact has been used in previous methods to partition the design into strips of width ϵ which are swept on the design, and the overlap is compensated. In FASCL, ϵ is an user defined parameter to the program. In SPACE [3], a finite element based capacitance extraction program, a scan based approach is used. A strip of width 2ϵ is moved from left to right over the layout in steps of ϵ, as a result of which neighbouring strips overlap half of their width ϵ. For each strip of width 2ϵ and for each overlap strip of ϵ, a coupling function is computed for all the nets in the strip. The computing function in this case is capacitance-extraction function based on hierarchal Schur's algorithm. A drawback of such scan based methods is that they assume the layout density to be uniform, which need not be the case. Typically, the chip layout is divided into a set of routing regions, which may be routing channels between rows of standard cells or global routing areas. Without information about the chip floorplan, scan based analysis tools would waste time analyzing regions where there are not many nets.

The primary objective of this paper is to describe a method of analyzing geometric proximate nets with respect to a predefined set of properties. If (and when) a property is satisfied by a *pair* of nets, an appropriate coupling evaluation function is called and the result of this function is added to the pairwise interaction quantum for the two nets in question. Our paper describes a geometric analyzer which computes a coupling function on pairs of nets closer than ϵ but is not based on a region based approach. In our opinion this is the first report of a method for *automatic application* of functions based on user defined rules, modes and properties. In addition we show that such an analysis need not be computationally intensive, even for designs which have large routing congestion in localized areas.

*Philips Research Labs, Eindhoven, The Netherlands. E-mail:{sandeep.koranne,o.p.gangwal}@philips.com.

The remainder of the paper is organized as follows. In the next section we give an example of a coupling function. In Section 3, we describe the architecture of FASCL. In Section 4, we describe our implementation and show results of experiments. Here we also describe how a typical application would be written using our framework. We conclude in Section 5.

2 Example of Coupling Function

The coupling function is a quantification of the pairwise interaction between nets. As an example, consider coupling capacitance. In a simple model, the coupling function depends on the following parameters; the coupling length between the two nets, the distance between them and the respective metal layers. In [7], the authors have assumed the coupling capacitance between neighbouring parallel wires to be given by the following formula:

$$C = \alpha \frac{coupling\ length}{distance^{\beta}} \tag{1}$$

where α is a technology dependent parameter and β is an experimentally determined constant. This function can easily be applied over all nets which are close in the layout using FASCL, as both *coupling length* and *distance* are known for all pairs of nets which are closer than ϵ. The aim of this paper is not to report on various extraction/analysis techniques, but rather to provide a generic framework in which such evaluating functions can be written and automatically applied. Hence, we do not consider the accuracy of the capacitance extraction (or for that matter, of other coupling functions, in this paper). We simply assume that a coupling function exists which returns a numeric value and takes parameters which are reported by the geometric kernel. More details on how to specify such a rule and function to FASCL along with what parameters are available for writing a rule, is provided in Section 4.

Since a long net is typically routed in many regions, and even on different metal layers, it is better to talk in terms of *segments*[1]. Each net comprises of one or more segments. Each segment is either horizontal or vertical, and is always routed on a single metal layer. We can write a net N_i to be a set of k segments, $s_{i1}, s_{i2}, \ldots, s_{ik}$. A segment s_p has a form (net_name $metal\_layer$ (x_1 y_1 x_2 y_2)), to represent the fact that it belongs to net *net_name*, is routed on metal layer *metal_layer* and has the given co-ordinates. The net *net_name* is called the *parent* of segment s_{pk}. The width of the metal wire (if it is available in the input), can also be used as a parameter to the function.

Using the above notation the coupling function between two nets p and q, $C_{p,q}$ is the sum of the coupling function of their segments. Instead of comparing all the segments of the two nets, we analyze two segments s_{px} and s_{qy}, which are close to each other ($distance_{p,q} < \epsilon$), and then add the coupling function value of the segments $C_{px,qy}$ to the value of their parents $C_{p,q}$. The geometric analysis kernel of FASCL is responsible for determining all pairs of nets which satisfy the proximity condition. The implementation of FASCL is shown in Sec. 3.

3 Architecture of FASCL

The architecture of FASCL is shown in Fig. 1. FASCL presents APIs to the user tool environment. Users can specify *modes* and *conditions* which are internally used by FASCL to apply the user specified functions on segments. FASCL itself comprises of the following parts :

Scan-line based geometrical kernel (SGK): The heart of FASCL is a scan-line based geometrical kernel, which signals an event whenever a pair of segments (polygon objects as analyzed from an abstract view of the layout) closer than ϵ are found. The parameter ϵ determines the accuracy versus run-time trade-off in FASCL.

Scan controller: The scan controller provides an interface to the SGK. Via the scan controller the user defined mode can specify ϵ. It is also possible to specify the start and stop ranges in the layout on which the scan is done.

Params Table: SGK is also responsible for maintaining the values of all[2] parameters listed in the *Params Table* for each interaction that it reports. This information is used to calculate the value of the user defined function and conditions.

Condition transform: This block is responsible for transforming the user defined condition, which is written by the user using specific *condition handles*, into a boolean value using the values from the Params Table.

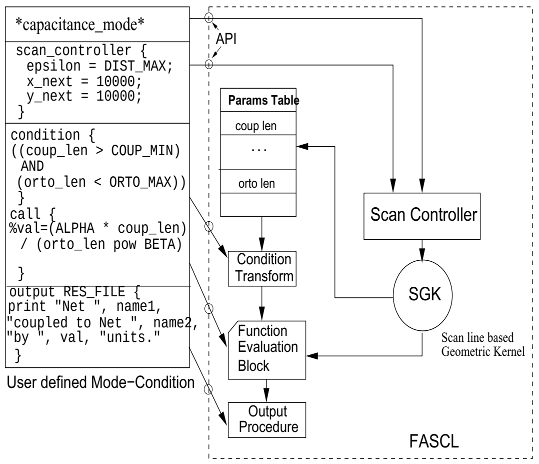
Function evaluation block: This block executes the user defined function whenever it receives

[1]We can replace *nets* by *segments* in the algorithm, as we assume that the *average-segmentation-factor* $Seg_{avg} = \frac{S_{total}}{No.\ of\ nets}$ is constant, and hence does not contribute to the asymptotic time complexity of computation.

[2]An optimizing post-processing step can remove unused items, which are not used in any user defined condition, from the Params Table.

an event from SGK and the boolean value from the condition transform block is TRUE.

Output procedure: The output of the function evaluation block is printed as defined by the user.



```
*capacitance_mode*
 scan_controller {
  epsilon = DIST_MAX;
  x_next = 10000;
  y_next = 10000;
 }
condition {
((coup_len > COUP_MIN)
 AND
 (orto_len < ORTO_MAX))
 }
call {
%val=(ALPHA * coup_len)
 / (orto_len pow BETA)

 }
output RES_FILE {
print "Net ", name1,
"coupled to Net ", name2,
"by ", val, "units."
 }
```

Figure 1: Architecture of FASCL

The input to the scanline based geometric kernel algorithm are two lists of segments and the current evaluation mode. The mode determines what conditions are checked, and the conditions in turn determine which functions are applied. The segment lists are sorted[3] by the y co-ordinate and the x co-ordinate, respectively. We describe the operation on one of these lists, as the operations on the other are similar in nature. The output of FASCL is the application of appropriate functions to all proximate nets which satisfy the currently active condition for that mode. The various parameters that a coupling function and conditions may use are shown in Table 1. The parameter TECH_PARAMS contain a collection of data specific to a particular technology. This may contain, for example, the capacitance to ground value for each metal layer expressed in term of unit length. It may also contain the unit resistance, mutual capacitance per metal layer. This information is used to calculate the coupling length when the two input segments are routed on different metal layers.

The pseudo-code for SGK is given below.

```
defglobal *active_set*;
procedure SGK;
input     segment_list;
begin
 dolist (x) in segment_list
  begin
    drop_old_items_from_active_set(x);
    update_params_table(x,active_set);
    insert_into_active_set(x);
    signal_event(x,active_set);
  end for
end
```

[3]Although the dependence of FASCL on a sorted list strictly requires a $O(S_{total} \log S_{total})$ time, for all practical circuits the preprocessing time for the sorting of the segment list is fairly small.

Table 1: Params Returned by Geometric Kernel

Name	Description
m1	Route layer for first seg.
m2	Route layer for second seg.
layer_diff	abs(m1-m2)
coup_len	Mutual coup. len
orto_len	Orthogonal dist. between first & second segment
max_m_layer	max(m1,m2)
min_m_layer	min(m1,m2)
len1	Len. of first seg.
len2	Len. of second seg.
nvias1	No. of vias in first net
nvias2	No. of vias in second net
shield val.	A function of no. of nets in-between the two segs
TECH_PARAMS	Technology parameters

There may exist a number of modes in FASCL, each with its own condition setting. The relation between `*curr-mode*` and `condition` has to be specified to the program. As an example consider the mode *CAPACITANCE*. The user can specify to the program the relation between the mode *CAPACITANCE* and its coupling condition which we denote **COUPORTO** as

```
COUPORTO=set_condition(*CAPACITANCE*)
begin
    ((coup_len > COUP_MIN) and
     (orto_len < ORTO_MAX))
end
```

The above construct tells the program that in *CAPACITANCE* mode for all pairs of segments the mutual coupling length should be more than COUP_MIN (units) and the orthogonal distance between the two segments should be less than ORTO_MAX. The last thing the user needs to tell the program is the name of the function that should be called when a certain condition is satisfied. This is done as :

```
set_function(COUPORTO) =
#'coupling_cap(coup_len, orto_len)
```

3.1 Analysis of FASCL

We present the analysis for the case of horizontal segments. Similar arguments apply for the set of vertical segments also. Let us assume that there are S_{total} segments. Then the `dolist` iteration in SGK is done S_{total} times. Both `drop_items_from_list` and `signal_event` take $O(b_i)$ time (although a more efficient implementation of `drop_items_from_list` can be written using binary trees, which would take time $O(\log b_i)$), where b_i is the number of *active_segments* `|*active_set*|`. The function `insert_into_list` takes constant time $O(1)$. The total run-time of the algorithm is then[4]:

$$T(S_{total}) = \sum_{i=1}^{S_{total}} [O(b_i) + O(\log b_i) + O(1)] \qquad (2)$$

It can be shown that FASCL will terminate in $O(S_{total}^{1+k})$ time where $k < 1$.

4 Implementation and Experimental Results

FASCL has been implemented in Common Lisp using Liquid Common Lisp Version 5.0.3 running on HP-UX A 9000/804 with 700MB RAM. The input to FASCL is a Cadence DEF file, which contains information about routing geometry for each net. A default rule set based on coupling length and substrate capacitance has been built. Since the program itself is written in Lisp, it is very easy to add new rules based on APIs provided by the program. The APIs support query information on parameters given in Table 1. This information can be saved in a database to be read in later, without the need for lengthy re-computation. It is possible to call other analysis programs like SPICE to analyze some net pairs.

Table 2: Computation Time for FASCL

Design	# Segments	Time(min)	Mem(MB)
D1	78681	0:04	20
D2	58769	0:14	20
D3	99217	0:30	53
D4	300486	0:46	118

As a typical example of the use of FASCL, we describe the creation of a *realistic bridge fault list* by FASCL, which was used to selectively analyze a subset of nets for bridge faults. This work is described in [4]. The program was tested with some industrial circuits. The circuits and results are described in Table 2. The circuit J1 was investigated for bridge faults. FASCL was used to generate a list of top 3 neighbours for a net. A set of patterns was generated which excited the victim and neighbour nets to different logic values (pairwise). The condition setting for the mode *BRIDGEFAULT* was :

```
COUPLEN=set_condition(*BRIDGEFAULT*)
begin
    ((layer_diff = 0) and (coup_len > 50))
end
```

It analyzes all pairs of segments which are coupled for more than 50 units (as read in from the DEF file) and are on the same metal layer.

[4]We are assuming that the coupling function is calculated in $O(1)$ time, w.r.t S_{total}.

5 Conclusions and Future Work

In this paper a framework for analysis of VLSI layouts has been described with the aim of automatic application of pairwise coupling functions to geometrically proximate nets. A unique mode-condition model has been implemented which facilitates the task of analyzing large designs. Theoretical analysis of the subquadratic run time of the program has been presented to show that this task is not very compute intensive. Experiments on circuits have demonstrated the usability and utility of the program. The method was used to generate a set of nets which represent a candidate set for bridge fault analysis. Future work will concentrate on providing more flexibility in the mode-condition model and a GUI.

References

[1] H. B. Bakoglu, *Circuits, Interconnections and Packaging for VLSI*, Addison Wesley, 1990.

[2] A. Devgan, “Efficient Coupled Noise Estimation for On-Chip Interconnects”, in *Proc. ICCAD*, 1997, pp. 147–153.

[3] A. J. van Genderen and N. P. van der Meijs, “Extracting Simple but Accurate RC Models for VLSI Interconnect”, in *Proc. ISCAS*, 1988, pp. 2351–2354.

[4] C. Hora, “Diagnosis of Bridging Defects in Random Logic”, *Workshop on Yield Optimization and Test, YOT 2000*, Oct. 2000.

[5] A. Vittal and M. Marek-Sadowska, “Crosstalk Reduction for VLSI”, *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, March 1997, vol. 16, pp. 290–298.

[6] T. Xue, E. S. Kuh and D. Wang, “Post Global Routing Crosstalk Synthesis”, *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, December 1997, vol 16, pp. 1418–1430.

[7] H. Zhou and D. F. Wang, “Global Routing with Crosstalk Constraints”, *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, November 1999, vol. 18, pp. 1683–1688.