

Interconnect Planning with Local Area Constrained Retiming*

Ruibing Lu and Cheng-Kok Koh
School of Electrical and Computer Engineering
Purdue University, West Lafayette, IN, 47907, USA
{lur, chengkok}@ecn.purdue.edu

Abstract

We present a framework that considers global routing, repeater insertion, and flip-flop relocation for early interconnect planning. We formulate the interconnect retiming and flip-flop placement problem as a local area constrained retiming problem and solve it as a series of weighted minimum area retiming problems. Our method for early interconnect planning can reduce and even avoid design iterations between physical planning and high level designs. Experimental results show that our method can reduce the number of area violations by an average of 84% in a single interconnect planning step.

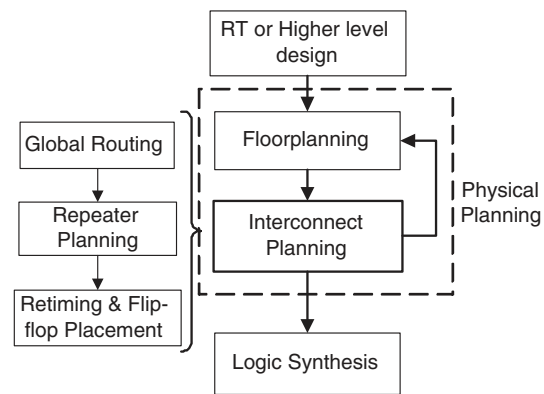


Figure 1. Interconnect Planning in Design Flow for Deep Submicron VLSI.

1 Introduction

Global interconnects not only dominate system performance, but also have a great impact on chip layout. While repeater planning [2, 11, 12, 1] can be used to consider the impacts of the large number of inserted repeaters on the floorplan, not all timing requirements (delay and transition time) can be met even by using repeaters. In fact, the wire delay can be as long as about ten clock cycles [9] in the near future; thus making pipelined signal transmission and the insertion of flip-flops or latches necessary.

Recent studies [10, 14, 7] began to incorporate into new design flows some flavor of physical design, called *physical planning*, during high level design stages. In the physical planning stage, floorplanning, global routing, repeater and flip-flop planning are performed to provide more accurate interconnect delay information to early design steps. The goal is to eliminate iterations of the entire design flow.

In [14], architectural level retiming with module selection was considered. Through architectural floorplanning, the clock cycles needed for each global interconnect was estimated. Then a new minimum area retiming problem was

formulated to deal with the area/delay tradeoff of circuit blocks under the clock cycle constraint. However, the paper did not consider the effects on the floorplan of the insertion of flip-flops and repeaters. These effects were considered in [7], in which repeaters and flip-flops were inserted simultaneously during interconnect planning. However, Ref. [7] did not consider how the latency of interconnects might affect the circuit behavior. Consequently, the planning result must be fed back to high level designs for proper adjustment to be made. Both of these two approaches relied on the iterations between the physical planning and high level design stages to achieve design convergence.

In this paper, we propose an interconnect planning framework that would reduce or even avoid iterations between high level design and physical planning. In this framework (shown in Figure 1), we not only take global routing and repeater planning into consideration, but also deal with *the retiming of logic blocks and interconnects and the placement of those flip-flops relocated by retiming*. The main advantage of our method is that correct timing and system behaviors are guaranteed; thus, the iterations between high level designs and physical designs can be avoided. If the placement of relocated flip-flops results in area con-

*This work was supported in part by NSF under contract number CCR-9984553 and a grant from Intel Corporation.

straint violations, iterations between the floorplanning stage and interconnect planning are required. The floorplanning stage could allocate additional space to those over-utilized soft blocks or channel regions for another iteration of interconnect planning.

Although our interconnect planning has three components as shown in Figure 1, the focus of this paper is on the third step because the other two have been addressed extensively in the literature. We formulate the retiming and flip-flop placement problem as a local area constrained retiming (LAC-retiming) problem, in which the retiming of both logic blocks and interconnects is performed with consideration of area constraints imposed by the floorplan. Here, the area constraints refer to the fact that the repeaters and flip-flops can be placed only in the soft blocks, dead areas and channel regions in the floorplan, and pre-located repeater/flip-flop sites [1] in hard blocks, subject to various capacity constraints. We solve the LAC-retiming problem by posing a series of weighted minimum area (min-area) retiming problems whose objective functions are adaptively adjusted. The experimental results show that in a single interconnect planning step the total number of area constraint violations can be reduced by 84% on the average. Except for one circuit, there are *no* area constraint violations after two iterations of interconnect planning.

2 Problem formulation

In this paper, we assume that the following information is available to the proposed interconnect planner:

- A register-transfer (RT) level netlist that describes the interconnections of RT level functional units (registers, multiplexers, ALU etc.).
- A partition of the RT level functional units into circuit blocks. These circuit blocks can be either hard or soft blocks. Hard blocks may have preallocated sites for repeater and flip-flop insertion [1].
- A floorplan of the circuit blocks; dead areas and channel regions in the floorplan can also be used for repeater and flip-flop insertion.
- The target clock period, denoted as T_{clk} .
- The maximum interval length between two consecutive repeaters, denoted as L_{max} . L_{max} is typically determined by the signal integrity constraint [1, 3].

The interconnect planner finds a solution that includes routing, repeater insertion for global (inter-block) interconnects, and retiming of logic and interconnects such that:

1. The total logic and interconnect delay between any two consecutive flip-flops is not larger than T_{clk} .

2. Repeaters and relocated flip-flops are placed in channel regions, dead areas, soft blocks, or repeater/flip-flop sites of hard blocks.
3. Wire length between any two consecutive repeaters is not larger than L_{max} .
4. Area constraint violation caused by the placement of repeaters and relocated flip-flops is minimized.

3 Retiming of logic and interconnects

Retiming [6, 13, 8, 14] performs sequential logic optimization by repositioning memory units. In this paper, we focus on synchronous circuits whose memory units are edge-triggered flip-flops. Insertion of flip-flops to break a long interconnect into two or more segments in order to meet the clock period constraint [7] can be viewed as retiming if those flip-flops are relocated from the nearby circuit blocks. In this paper, only minimum area (min-area) retiming is of interest since our goal is to minimize the area constraint violations while meeting the clock period constraint. The objective of min-area retiming is the minimization of the total area of flip-flops under a given clock period constraint. In the following discourse, we first review min-area retiming, and then present the concept of interconnect retiming.

3.1 Minimum area retiming

A sequential circuit can be represented by a weighted directed acyclic graph $G(V, E)$, where each vertex v corresponds to a functional unit v , and the weight of v is the delay of that functional unit. The directed edges E model the interconnects between the functional units, and the weight $w(e_{u,v})$ of edge $e_{u,v}$ is the number of flip-flops along the connection between functional units u and v . The fanouts and the fanins of vertex u are denoted by $FO(u)$ and $FI(u)$, respectively. Retiming can be viewed as a labeling of vertices $r : V \rightarrow Z$ where Z is the set of integers. The weight of edge after retiming $w_r(e_{u,v})$ is defined by the equation

$$w_r(e) = w(e_{u,v}) + r(v) - r(u).$$

There are two kinds of constraints for min-area retiming:

1. The edge weights cannot be negative. Thus:

$$r(v) - r(u) \geq -w(e_{u,v}), \forall e_{u,v} \in E. \quad (1)$$

2. For any path, $u \rightsquigarrow v$, with path delay larger than clock period T_{clk} , there should be at least one flip-flop on it:

$$r(v) - r(u) \geq -W(u, v) + 1, \forall u \rightsquigarrow v, D(u, v) > T_{clk}, \quad (2)$$

where $W(u, v)$ defines the minimum latency for signal to transfer from u to v and $D(u, v)$ is the maximum delay from u to v under the minimum latency.

The objective of min-area retiming is to reduce the number of flip-flops, if all flip-flops are assumed to have unit area. It can be shown that the objective function is:

$$\begin{aligned} N(G_r) &= \sum_{e_{u,v} \in E} w_r(e_{u,v}) \\ &= \sum_{e_{u,v} \in E} w(e_{u,v}) + \sum_{v \in V} r(v)(|FI(v)| - |FO(v)|). \end{aligned}$$

In this equation, the first term is a constant. Therefore, the second term becomes the objective function of min-area retiming. As every constraint of min-area retiming is a difference of two retiming variables, min-area retiming can be solved efficiently by transforming it into the minimum-cost flow problem [6].

3.2 Retiming of interconnects

In deep submicron VLSI circuits, the delay of global interconnects can no longer be ignored. In fact, the delay of some interconnects is so significant that they should be broken into two or more segments by flip-flops in order to meet the clock period constraint. However, flip-flop insertion alters the system behavior. In order to maintain the correct system behavior, we should relocate flip-flops from the logic units into the interconnects; this is exactly what retiming does.

Traditional min-area retiming considers only functional units; it completely ignores the interconnects. To render traditional retiming applicable to interconnects, we represent each interconnect as a series of *interconnect units*, which have delay but perform no logic function. Repeater insertion provides a natural segmentation of an interconnect into interconnect units, with the delay of each unit being the sum of the repeater delay and the delay of the interconnect segment driven by the repeater.

Even more flexibility can be introduced if we further divide the interconnect segment between two repeaters into several interconnect units. However, this presents some problem for retiming. For ease of retiming, all units of a netlist should have fixed delays, but the introduction of flip-flops in the middle of an interconnect would change the delay of fanin and fanout interconnect segments. An approach around this problem is to find out the maximum delay of an interconnect segment under all possible ways of inserting flip-flops and assign that delay to the segment. The drawback is that the accuracy of interconnect delay is sacrificed.

A more significant problem with interconnect retiming is that flip-flops consume chip area. Although min-area retiming is optimal in terms of overall area consumption, it may

relocate flip-flops from regions with a lot of empty space to over-utilized regions in order to minimize the total area consumption. That will render the given floorplan invalid because of the area constraint violations, and necessitate iterations of floorplanning and interconnect planning (See Figure 1). This is highly undesirable because minimizing such iterations is our primary objective.

Therefore, for interconnect retiming, we formulate a new retiming problem in which both the timing and the impact on floorplan of the relocated flip-flops are considered. We refer to the new formulation the local area constrained retiming problem and present a solution to it in the next section.

4 Interconnect planning with LAC-retiming

Interconnect planning has three tasks: global routing, repeater planning, and retiming. In this paper, we focus on the LAC-retiming and flip-flop placement problem.

In order to consider the effects on floorplan of repeaters and relocated flip-flops, we divide the chip layout into tiles. We handle the tiles differently as follows:

- Tiles in the channel regions, dead areas of the floorplan or hard blocks. Typically tiles in channel regions and dead areas have high capacity for repeater and flip-flop insertion. The tiles in hard blocks typically have very low additional area capacity unless some repeater and flip-flop sites [1] are inserted intentionally.
- Tiles in soft circuit blocks. Since the layout of soft blocks are not performed, we assume that as long as the total area consumption of functional units, repeaters and flip-flops in a soft block is not larger than its capacity, the layout of this block can be finished in the placement stage. Therefore, we merge all the tiles in a soft block together. The capacity of this merged block tile is the difference between the total capacity and the area consumed by its functional units.

In the rest of this paper, the word “tile” refers to either a regular tile or a merged soft block tile. Figure 2 shows a tile graph with channel regions, hard blocks and soft blocks.

4.1 Global routing and repeater planning

The first step of our approach establishes the global routing so that accurate estimation of delay and area consumption of global interconnects in subsequent steps can be obtained. In this step, the reduction of wire length and the routing congestion is the primary objective. A secondary objective considered is the possible area consumption due to future repeater/flip-flop insertion. Any time-driven and congestion-aware global router can be used for this step.

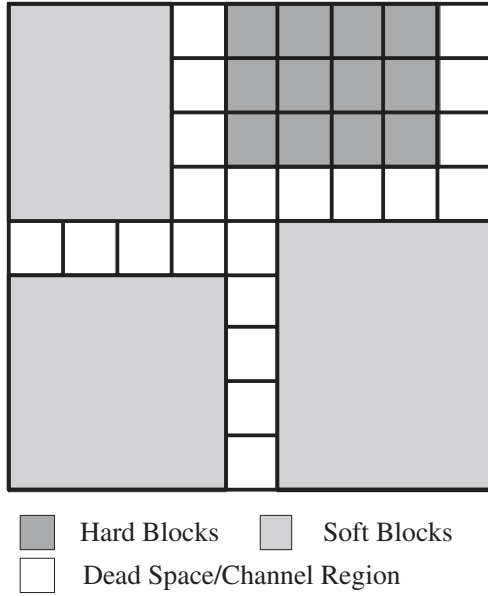


Figure 2. Tile graph for LAC-Retiming

We adapt the algorithm from [5] for the steiner tree construction. We also apply rip-up and re-routing to reduce routing congestion.

In the proposed interconnect planning steps, we perform repeater planning based on maximum interval length constraint L_{max} . This is the constraints defined based on a desirable signal integrity level [1, 3, 4, 12]. We use the dynamic programming repeater insertion algorithm from [1].

Note that both the routing and repeater insertion performed are primary for area and delay estimation. The result can also be used to guide the final routing and repeater insertion in the later physical design steps when more detailed timing information are available.

4.2 Local area constrained retiming

This step performs retiming of both logic units and interconnects under the clock period constraint. The objective is to minimize area constraint violations, so that the number of iterations between floorplanning and interconnect planning can be reduced. During the retiming, the area consumption due to relocated flip-flops need to be computed. Since the relationship between the functional units and circuit blocks in the floorplan is known, the position of a flip-flop can be estimated based on its fanin/fanout functional units or interconnect units. To ease the computation, we assume that each flip-flop is placed in the same tile as its fanin functional unit or interconnect unit.

Now we define the local area constraints formally. Let T be the set of all tiles, and for any $t_i \in T$, $C(t_i)$ be its capacity for flip-flop insertion. Note that this is the remaining capac-

ity after repeater insertion. We define a function $P : V \rightarrow T$ that maps each functional unit $v \in V$ to a tile $t_i \in T$. In other words, $P(v) = t_i$ means that functional unit or interconnect unit v is in the tile t_i of the floorplan. Therefore, the area constraint of a tile requires require that

$$\sum_{P(u)=t_i, e_{u,v} \in E} (w(e_{u,v}) + r(v) - r(u)) \leq C(t_i), \quad \forall t_i \in T. \quad (3)$$

The LAC-retiming problem is to find a retiming vector such that edge weight constraints (Eqn. (1)), the clocking constraints (Eqn. (2)) and local area constraints (Eqn. (3)) are satisfied. Even though the local area constraints are linear in terms of the retiming vector, the LAC-retiming problem cannot be transformed into minimum-cost flow problem because each local area constraint involves more than two retiming variables. Rather, it is a integer linear programming problem, which is NP-Complete.

We propose a heuristic based on min-area retiming to solve the LAC-retiming problem. In the min-area retiming, it is assumed that all flip-flops have the same area cost; thus, the minimization of total number of flip-flops can guarantee the minimum total area consumption. In the LAC-retiming, the cost of inserting flip-flops into different tiles are no longer the same because the tile capacities may be different. Therefore, we assign different weights to flip-flops in different tiles, and adaptively adjust them based on the area consumption and tile capacities. The different weights can guide the min-area retiming algorithm to reposition flip-flops from over-utilized tiles to those with low area consumption.

We define the weighted min-area retiming as follows: Function $A : V \rightarrow R^+$ assigns area weights to all functional units and interconnect units. And the weighted fanin/fanout costs are defined as:

$$fi(v) = \sum_{u \in FI(v)} A(u),$$

$$fo(v) = A(v)|FO(v)|, \forall v \in V.$$

The objective of weighted min-area retiming becomes:

$$N'(G_r) = \sum_{e_{u,v} \in E} A(u) \cdot w_r(e_{u,v})$$

$$= \sum_{e_{u,v} \in E} A(u) \cdot (w(e_{u,v}) + r(v) - r(u))$$

$$= N'(G) + \sum_{v \in V} r(v)(fi(v) - fo(v)),$$

where $N'(G)$ is constant. Therefore, the second term is the objective function to be minimized in the weighted min-area retiming. This objective function is in the same form as that of min-area retiming. Thus, the weighted min-area retiming can be solved effectively by using any minimum-cost flow algorithm.

The overall algorithm for LAC-retiming is as follows:

1. Get the edge weight constraints (Eqn. (1)) and clocking constraints (Eqn. (2)).
2. Assign uniform weight to all functional units and interconnect units.
3. Solve the weighted min-area retiming problem.
4. Compute the area consumption $AC(t_i)$ for each tile t_i in T .
5. If area consumption for every tile t_i is not larger than its capacity $C(t_i)$, or the result is not improved for N_{max} times, exit.
6. Assign new weight to each tile based on the LAC-retiming result obtained in step 3:

$$\text{New Tile Weight} = \text{Previous Tile Weight} \times \left\{ (1 - \alpha) + \alpha \times \frac{AC(t_i)}{C(t_i)} \right\}.$$

Assign this new weight to all functional and interconnect units in this tile. Then, goto step 3.

The algorithm terminates either when all local area constraints are met or when there is no improvement after some pre-specified number (N_{max}) of consecutive iterations. The coefficient α ($0 \leq \alpha \leq 1$) is used to adjust the influence of the previous weight on the new weight. Experimental results indicated that a value of around 0.2 typically produces the best results.

Although our heuristic perform a series of weighted min-area retiming, the time complexity is not as high as it seems. As pointed out in [8], a significant portion of the total execution time of min-area retiming is spent on computing the clocking constraints. In contrast, solving the minimum-cost flow problem is known to be quite efficient. In our heuristic, the clock period constraints are generated only once. Furthermore, it typically takes only a few iterations to get the final result. Therefore, the time complexity of this heuristic is in the same order as that of min-area retiming. This is further validated by our experimental results.

5 Experimental results

We performed our experiment on some ISCAS89 benchmark circuits. Although those circuits are gate level netlists, for the lack of RT level benchmarks, we treat them as RT level netlist, i.e., we assume that the gates in those netlists are functional units with large area and delay. In order to obtain proper inputs for our algorithm, we first partition those circuits into soft blocks and use a sequence pair floorplanner to compute the floorplan. We then perform routing and repeater planning. We use T_{init} to denote the smallest clock

cycle under which the circuit can operate at this stage, i.e., after floorplanning, routing, and repeater insertion. After that, we perform min-period retiming, without consideration of area constraints, to obtain the minimum clock period T_{min} . Finally, both LAC-retiming and min-area retiming are performed respectively to get two sets of experimental results. We set the target clock period, denoted by T_{clk} , for both min-area retiming and LAC-retiming to be between T_{init} and T_{min} ; the difference between T_{clk} and T_{min} is 20% of the difference between T_{init} and T_{min} .

The experimental results are shown in Table 1. " N_{FOA} " is the total number of flip-flops that violate the local area constraints. For those circuits whose N_{FOA} cannot be reduced into zero in one iteration of interconnect planning, we also provide in parenthesis N_{FOA} of the second iteration of interconnect planning after the incremental change of the floorplan (see discussion below). N_F and N_{FN} are the total number of flip-flops and the number of flip-flops inserted into interconnects, respectively. " N_{wr} " is the number of weighted min-area retiming performed to solve the LAC-retiming. " T_{exec} " denotes the run time of the retiming part. The percentage decrease of N_{FOA} through LAC-retiming is shown as " N_{FOA} Decr."

The results shows LAC-retiming can decrease the total number of area violation by 84% at the expense of a possible slight increase in the total number of flip-flops. Except for three circuits, the area violations can be completely removed. Even for these three circuits, the amount of area violations is greatly reduced. The reason for these area violations is that the area consumption of each block is based on the original netlist without any physical information in the first iteration of physical planning.

For the three circuits with area violations, we expand those congested soft blocks and channel, and then perform another iteration of interconnect planning. Except for circuit s1269, all the area constraint violations are completely removed. In the case of circuit s1269, the target clock cycle becomes infeasible after the floorplan expansion. The reason is that the amount of area violation is so large that the floorplan changes drastically. This further proves the necessity to reduce the area violations as many as possible. The reduction of area constraint violations would speed up the convergence of the physical planning step even if not all area constraint violations can be removed. Through experiments, we also found that the results for circuit s1269 can be improved greatly by changing the circuit partition. Therefore, we expect better convergence of the physical planning step if an approach considering both partition and floorplanning is used.

We also observe from the experimental results that, on the average, about 10% of the flip-flops are inserted into interconnects; the percentage can be as high as 30%. For some circuits, there is a large difference between the ini-

Table 1. Experimental Results

circuit	T_{clk} (ns)	T_{init} (ns)	Min-Area Retiming				LAC-Retiming					N_{FOA} Decr.
			N_{FOA}	N_F	N_{FN}	T_{exec} (s)	N_{FOA}	N_F	N_{FN}	N_{wr}	T_{exec} (s)	
s1196	22.1	22.1	0	18	0	0.2	0	18	0	1	0.2	N/A
s1269	11.5	16.5	22	69	11	0.6	16 (N/A)	77	17	7	3.2	27%
s3330	27.5	30.7	5	79	23	2.0	0	79	29	2	2.9	100%
s3384	31.6	38.8	20	157	7	3.1	5 (0)	163	34	8	12.9	75%
s4863	25.4	28.1	5	94	0	48.7	0	94	5	2	64.3	100%
s5378	42.9	50.1	23	156	23	17.3	0	164	20	2	24.4	100%
s6669	42.5	90.8	4	203	12	18.3	0	204	37	4	42.5	100%
s9234.1	42.1	48.5	4	150	2	23.4	2 (0)	150	8	5	64.8	50%
s13207.1	108.5	108.5	8	476	27	84.7	0	476	54	2	95.7	100%
s15850.1	131.7	162.5	3	527	22	157.4	0	527	46	2	167.6	100%
Average												84%

tial clock period and minimum clock period. That is caused by the unbalanced distribution of flip-flops as the interconnect delay cannot be estimated accurately at high level design steps. Both observations also demonstrate the necessity of retiming and flip-flop relocation during the interconnect planning step.

The execution time of LAC-retiming is in the same order as that of min-area retiming, and only a few calls to the weighted min-area retiming are required to achieve final results. In our implementation of LAC-retiming, the clock period constraint generation is based on the algorithm from [13]. This implementation can be improved by the constraint reduction technique in [8]. Since that technique can significantly reduce the run time of the minimum-cost flow problem, and our approach executes minimum-cost flow algorithm iteratively, we expect significant reductions in execution times by using that technique.

6 Conclusion

In this paper, we have presented a framework for interconnect planning that includes global routing, repeater planning, flip-flop relocation and placement through LAC-retiming. The design iterations between physical planning and high level designs typically can be avoided through this method. Experimental results show that our technique can reduce the number of area constraint violations by 84% on the average in one iteration of interconnect planning.

References

[1] C. J. Alpert, J. Hu, S. S. Sapatnekar, and P. G. Villarrubia. A practical methodology for early buffer and wire resource allocation. In *Proc. Design Automation Conf.*, pages 189–193, 2001.

[2] J. Cong, T. Kong, and D. Z. Pan. Buffer block planning for interconnect-driven floorplanning. In *Proc. Int. Conf. on Computer Aided Design*, pages 358–363, 1999.

[3] F. Dragan, A. Kahng, I. Mandioui, S. Muddu, and A. Zelikovsky. Provably good global buffering using an available buffer block plan. In *Proc. Int. Conf. on Computer Aided Design*, page 104, 2000.

[4] F. Dragan, S. Muddu, E. Sarto, and R. Sharma. Interconnect tuning strategies for global interconnects in high-performance deep-submicron ics. In *Design, Automation and Test in Euro. Conf.*, 1998.

[5] J. M. Ho, G. Vijayan, and C. K. Wong. New algorithms for the rectilinear Steiner tree problem. *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, 9(2):185–193, 1990.

[6] C. E. Leiserson and J. B. Saxe. Retiming synchronous circuitry. *Algorithmica*, 6:87–116, 1991.

[7] R. Lu, G. Zhong, C.-K. Koh, and K.-Y. Chao. Flip-flop and repeater insertion for early interconnect planning. In *Design, Automation and Test in Euro. Conf.*, page 690, 2002.

[8] N. Maheshwari and S. S. Sapatnekar. An improved algorithm for minimum-area retiming. In *Proc. Design Automation Conf.*, 1997.

[9] D. Matzke. Will physical scalability sabotage performance gains? *IEEE Computer*, 8:37–39, Sept. 1997.

[10] R. H. Otten and R. K. Brayton. Performance planning. *Integration, the VLSI Journal*, 29:1–24, 2000.

[11] P. Sarkar and C.-K. Koh. Routability-driven repeater block planning for interconnect-centric floorplanning. In *Proc. Int. Symp. on Physical Design*, pages 186–191, 2000.

[12] P. Sarkar and C.-K. Koh. Repeater block planning under simultaneous delay and transition time constraints. In *Design, Automation and Test in Euro. Conf.*, pages 540–544, 2001.

[13] N. Shenoy and R. Rudell. Efficient implementation of retiming. In *Proc. Int. Conf. on Computer Aided Design*, pages 226–233, 1994.

[14] A. Tabara, B. Tabbara, R. K. Brayton, and A. R. Newton. Integration of retiming with architectural floorplanning. *Integration, the VLSI Journal*, 29:25–43, 2000.