Performance Metric Based Optimization Protocol

Xavier Michel, Alexandre Verle, Philippe Maurine, Nadine Azemard, Daniel Auvergne

Performance Metric Based Optimization Protocol

X. Michel, A. Verle, P. Maurine, N. Azémard, and D. Auvergne

LIRMM, UMR CNRS/Université de Montpellier II, (C5506),
161 rue Ada, 34392 Montpellier, France
{azemard,pmaurine,auvergne}@lirmm.fr

Abstract. Optimizing digital designs implies a selection of circuit implementation based on different cost criteria. Post-processing methods such as transistor sizing, buffer insertion or logic transformation can be used for optimizing critical paths to satisfy timing constraints. However most optimization tools are not able to select between the different optimization alternatives and have high CPU execution time.
In this paper, we propose an optimization protocol based on metrics allowing to characterize a path and to select the best optimization alternative. We define a way to characterize the design space of any circuit implementation. Then we propose a constraint distribution method allowing constraint satisfaction at nearly minimum area. This quasi optimal tool is implemented in an optimization tool (POPS) and validated by comparing the area necessary to satisfy delay constraints applied to various benchmarks (ISCAS'85) to that resulting from an industrial tool.

1 Introduction

Trade-off between speed, power and area can be achieved with circuit simulators and critical path analysis tools to modify iteratively the size of the transistors until complete constraint satisfaction [1-4]. More general speed-up techniques involve buffer insertion [5-6] and logic transformation [7]. If these techniques may be found efficient for speeding-up combinational paths they may have different impacts in the resulting power dissipation or area. Gate sizing is area (power) expensive and, due to the resulting capacitive loading effects, may slow down adjacent upward paths. This implies complex and iterative timing verifications. Buffer insertion preserves path interaction but is only efficient for relatively highly loaded nodes. To manage these alternatives it is necessary to evaluate and compare the performance of the different implementations. Without using any robust indicator, selecting between all these different techniques for the various gates of a library is NP complex and induces more iterative attempts which are processing time explosive.

A reasonable selection of speed-up technique must be based on a characterization of the available speed on a critical path, on the determination of the critical nodes and the characterization of the gate sensitivity to the sizing or buffering alternatives.

The main contribution of this paper is to define different metrics for path characterization, transistor sizing and buffer insertion, to be used as efficient indicators for characterizing the logic gates in terms of sensitivity to the sizing and buffering techniques.

E. Macii et al. (Eds.): PATMOS 2004, LNCS 3254, pp. 100–109, 2004.

Section 2 presents the elements used to define the optimization protocol. The optimization alternative with structure conservation is presented and validated in section 3. The proposed optimization method with buffer insertion is detailed and validated in section 4, in which the resulting optimization protocol is presented, before to conclude in section 5.

2 Optimization Protocol

Current path optimization tools [8] require large CPU times and too significant calculation computer resources to manage the complexity of nowadays developed circuits [9]. The uncertainty in parasitic capacitance estimation imposes to use many iterations or to consider very large safety margin resulting in oversized circuits.

2.1 Optimization Tool

As a solution to these drawbacks, we have developed an analysis and performance optimization tool based on an accurate representation of the physical abstraction of the layout (POPS: Performance Optimization by Path Selection) [10]. It gives facilities in analyzing and optimizing combinatorial circuit paths in submicronic technologies.

This tool allows to consider an user specified limited number of paths [11-12], for easy application and validation of the different path optimization criteria. The delay model implemented in this tool is based on an analytical representation of the timing performance, allowing to obtain for any logic gate, in its environment, an accurate evaluation of its switching delay and output transition time.

2.2 Delay Model

Real delay computation must consider finite input transition and I/O coupling [13]. We capture the effect of the input-to-output coupling and the input slope effect in the delay as

$$t_{HL}(i) = \frac{v_{TN}}{2}\tau_{INLH}(i-1) + \left(1 + \frac{2C_M}{C_M + C_L}\right)\frac{\tau_{outHL}}{2}(i)$$
$$t_{LH}(i) = \frac{v_{TP}}{2}\tau_{INHL}(i-1) + \left(1 + \frac{2C_M}{C_M + C_L}\right)\frac{\tau_{outLH}}{2}(i) \quad (1)$$

where $\tau_{INHL,LH}$, $\tau_{outHL,LH}$ are the input and output transition time duration, respectively. C_M is the coupling capacitance between the input and output nodes, that can be evaluated as one half the input capacitance of the P(N) transistor for input rising (falling) edge, respectively or directly calibrated from SPICE simulation.
The general expression of the transition time has been developed in [14] as

$$\tau_{outHL} = \tau \cdot S_{HL} \cdot \frac{C_L}{C_{IN}}$$
$$\tau_{outLH} = \tau \cdot S_{LH} \cdot \frac{C_L}{C_{IN}} \quad (2)$$

$$S_{HL} = (1 + k) \cdot DW_{HL}$$
$$S_{LH} = R \cdot \frac{(1 + k)}{k} \cdot DW_{LH} \quad (3)$$

where τ is a time unit that characterizes the process. C_L, and C_{IN} represent, respectively, the output load and the gate input capacitance. $S_{HL,LH}$ represent the symmetry factor of the falling, rising edges. R represents, for identical load and drive capacitance, the ratio of the current value available in N and P transistors, k is the P/N configuration ratio and $D_{WHL,LH}$ the gate logical weight defined by the ratio of the current available in an inverter to that of a serial array of transistors [14].

If eq.2,3 are quite similar to the logical effort expressions [4], they only represent the transition time expression. The delay is given by (1) that completely captures the input-to-output coupling and the input transition time effect on the delay. Using these expressions to define metrics for optimization, we always consider that the resulting implementation is in the fast input control range [14].

As shown from eq. (1-3) the delay on a bounded combinatorial path is a convex function and these expressions can easily be used to determine the best condition for path optimization under delay constraint.

By bounded combinatorial path we signify that the path input gate capacitance is fixed by the load constraint imposed on the latch supplying the path. This implies that the path terminal load is completely determined by the total input capacitance of the gates or registers controlled by this path. This guarantees the convexity of the delay on this path.

3 Optimization with Structure Conservation

The goal of gate sizing is to determine the optimum size for path delay constraint satisfaction at the minimum area/power cost. For that an essential parameter to be considered is the feasibility of the constraint imposed on the path. The target of this section is twofold: defining the delay bounds of a given path and determining a way for distributing a delay constraint on this path with the minimum area/power cost.

3.1 Constraint Feasibility

This is the important section of this approach. Without indication on the feasibility of a constraint any iterative method may infinitely loop with no chance to reach a solution. For that, in order to verify the feasibility of a constraint, we explore the path optimization space by defining the max and min delay bounds (Tmax, Tmin) of this path. It is clear that if the delay constraint value is lower than the minimum delay achievable on this path, whatever is the optimization procedure, there is no way to satisfy the constraint without path modification. These bounds are of great importance in first defining the optimization alternative.

Theoretically and without gate size limitation, no upper delay bound can be defined a path. To define a pseudo-upper bound we just consider a realistic configuration in which all the gates are implemented with the minimum available drive.

The definition of the lower bound has been the subject of numerous proposals. For ideal inverters without parasitic loading the minimum is reached when all the inverters have an equal tapering factor that can be easily calculated from a first order delay representation [7,15]. Applying the explicit representation given in (1) to a bounded combinatorial path, the inferior delay bound is easily obtained by canceling the de-

rivative of the path delay with respect to the input capacitance of the gates. This results in a set of link equations

$$C_{IN}^{2}(i) = \frac{A_i}{A_{i-1}} \cdot \left(C_{IN}(i+1) + C_{par}(i)\right) \cdot C_{IN}(i-1) \quad (4)$$

where (i) specifies the rank of the gate, $C_{par}(i)$ is the gate (i) output parasitic capacitance and the A_i correspond to the design parameters involved in (1,2).

As shown, the size of gate (i) depends on that of (i+1) and (i-1). This is exactly what we are looking for. Instead to solve the corresponding set of equations we prefer to use an iterative approach starting from a local solution defined with $C_{IN}(i-1)$ equal to the minimum available drive (C_{REF}). Then processing backward from the output, where the terminal load is known, to the input, we can easily determine an initial solution. Then by applying this solution in (4) we can reach, after few iterations, the minimum of delay achievable on the path. An illustration of the evolution of these iterations is given in Fig.1. We can easily verify that the final value, tmin is conserved whatever is the initial solution, ie the C_{REF} value.

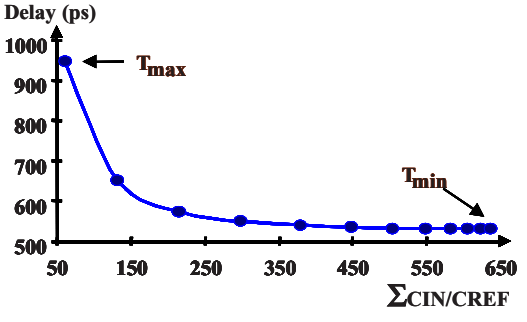

Fig. 1. Illustration of the sensitivity of the path delay to the gate sizing

This method has been implemented in POPS. Validation has been obtained by comparing on the longest path of different ISCAS'85 benchmarks (process CMOS, 0.25µm) the minimum delay value, obtained from the proposed method, to that reached by an industrial tool (AMPS from Synopsis). Fig.2 illustrates the resulting comparison that demonstrates the accuracy of the proposed method.

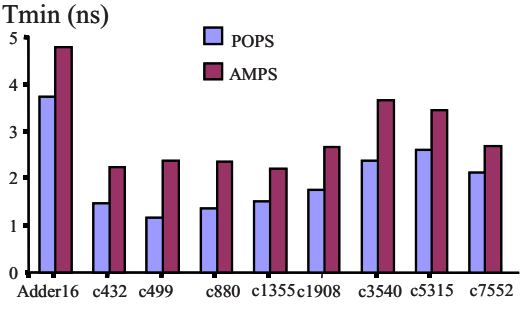

Fig. 2. Comparison of the minimum delay value (Tmin) determined with POPS and AMPS.

For any path the determination of the delay bounds gives facilities in verifying the feasibility of the constraint. For a delay constraint value higher than the minimum bound, the optimization alternative to be chosen is transistor sizing with structure conservation. Next step is to develop a fast technique allowing to efficiently distribute the constraint on the path.

3.2 Constraint Distribution: Constant Sensitivity Method

Several methods can be used. The simplest method is the Sutherland method [4], directly deduced from the Mead's optimization rule of an ideal inverter array [15]: the same delay constraint is imposed on each element of the path. If this supplies a very fast method for distributing the constraint, this is at the cost of an over sizing of the gates with an important logical weight value

We propose a new method based on the gate sensitivity to the sizing, that can be directly deduced from (4), as illustrated in Fig.3.

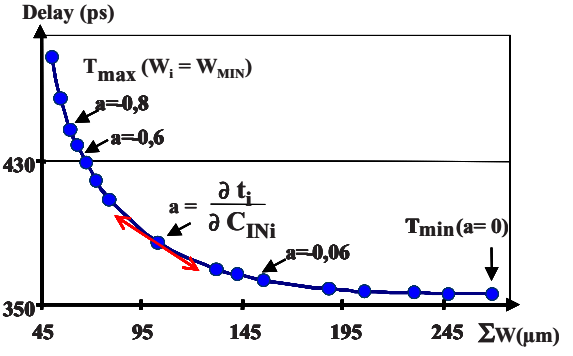

Fig. 3. Example of design space exploration on a 11gate path, using the constant sensitivity method.

This Figure represents the variation of the path delay to the gate sizing. Each point has been obtained by imposing the same value of each partial derivative:

$$\frac{\partial T}{\partial C_{IN}(i)} = a \tag{5}$$

"a" = 0 corresponds to the minimum, varying the value of this coefficient from 0 to a high negative value allows the exploration of the full design space
Solving:

$$\begin{aligned} A_{i-1} \cdot \frac{1}{C_{i-1}} - A_i \cdot \frac{C_{i+1} + C_{Pi}}{C_i^2} &= a \\ A_i \cdot \frac{1}{C_i} - A_{i+1} \cdot \frac{C_{i+2} + C_{Pi+1}}{C_{i+1}^2} &= a..... \end{aligned} \tag{6}$$

supplies a sizing solution for each value of the sensitivity coefficient "a", at which corresponds a value of the path delay. Few iterations on the "a" value allows a quick satisfaction of the delay constraint.

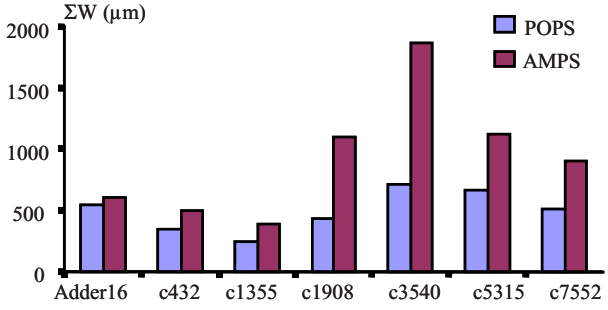

Fig. 4. Comparison of the constraint distribution methods on different ISCAS circuits.

Table 1. CPU time comparison in satisfying path delay constraint.

Circuits	Gate nb	POPS (ms)	AMPS (ms)
Adder16	99	159	23700
fp	14	19	6120
c432	29	29	9950
c499	29	30	9050
c880	28	29	9850
c1355	30	49	11400
c1908	44	49	11760
c3540	58	69	15890
c5315	60	90	19400
c6288	116	210	21920
c7552	47	69	16400

This method has been implemented in POPS and validated on different ISCAS circuits. In Fig.4 we compare the final area, given as the sum of the transistor widths (ΣW), necessary to implement the critical path of each circuit under an identical hard constraint (Tc = 1.2Tmin), using POPS and AMPS. As shown the equal sensitivity method results in a smaller area/low power implementation.

In Table 1 we compare the CPU time necessary for AMPS and POPS in sizing under delay constraint different benchmarks. As illustrated the use of a deterministic approach in POPS, results in a two order speed up of the constraint distribution step, compared to the random approach used in standard tools (AMPS).

When the delay constraint has a smaller value than the minimum delay available, the only alternative is to modify the structure of the path.

4 Optimization with Buffer Insertion

The goal of this part is to define a way to select between sizing and buffer insertion. We just focus here on the buffer insertion method, that can be easily extended to the logic path modification. The problem is to determine, at minimum area cost, the best location to insert a buffer and the minimal sizing satisfying the delay constraint.

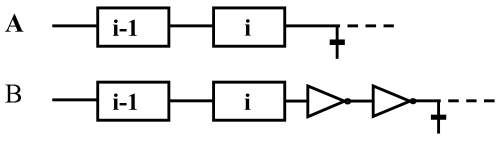

Fig. 5. Local buffer insertion

In Fig 5 we represent a path general situation where an overloaded node is guessed to be sped up by buffer insertion. The problem is to remove the guess by a metric directly determining for what level of load a gate switching speed can be improved. For that we compare the delay (1) of the A and B structures for determining at what fan out value ($F = C_L/C_{IN}$) the B structure becomes faster than A. This defines the "load buffer insertion limit" (Flimit). In a first step we use a local insertion method in which we conserve the size of gates (i-1) and (i) and just size the buffer (4), for minimizing the delay between the output of (i) and the terminal load.

The values of these Flimit are listed in Table 2. In the configuration of Fig.5, (i-1) is an inverter and we have considered the evolution of the limit with the gate (i). A complete characterization must involve all possibility of (i-1) gates and can be done easily following the same procedure. Validation of these limits has been obtained through Hspice simulations. As expected, greater is the logical weight of the gate, lower is the limit that may constitute a measure of the gate efficiency.

In fact the buffer insertion acts as a load dilution for the initial gate. In this case the size of this gate can be decreased. A complete consideration of the method involves using the predefined limits for critical nodes identification and then to distribute the delay constraint on the full path using the constant sensitivity method in order to preserve an area efficient gate sizing.

Validation of this approach is given in Table 3 where we compare the minimum delay obtained, from POPS, on the different ISCAS circuits using sizing and buffer insertion techniques. As shown, depending on the path structure significant minimum delay value improvement can be obtained with buffer insertion. Note that considering the delay sensitivity to the gate sizing (Fig.4), any minimum delay improvement on a path will result in a delay constraint satisfaction with smaller area.

This is illustrated in Fig.6 were we compare, on a 13 gate array, the path delay versus the area for the two methods: gate sizing (full line) and buffer insertion with global gate sizing (dotted line).

Three regions can be defined, a weak constraint domain where sizing is the best solution ($Tc > 2.5Tmin$), a medium constraint domain where buffer insertion is not necessary, but allows path implementation with area reduction ($1.2Tmin < Tc < 2.5Tmin$) and a hard constraint domain ($Tc < 1.2Tmin$), where buffer insertion is the

Table 2. Fan out limit (Flimit) for a gate (i) controlled by an inverter.

Gate i-1	Gate i	Calcul.	Simulation
inv	inv	5,7	5,9
inv	nand2	4,9	5,4
inv	nand3	4,5	5,2
inv	nor2	3,8	3,5
inv	nor3	2,7	2,5

Table 3. Comparison of sizing and buffer insertion techniques.

Circuits	Method	Tmin(ns)
Adder	sizing	4,53
	buff	4,39
	gain	3%
c432	sizing	2,22
	buff	1,97
	gain	13%
c499	sizing	1,79
	buff	1,64
	gain	9%
c880	sizing	2,09
	buff	1,71
	gain	22%
c1355	sizing	2,16
	buff	1,89
	gain	14%

Circuits	Method	Tmin(ns)
c1908	sizing	2,66
	buff	2,32
	gain	15%
c354	sizing	3,29
	buff	3,21
	gain	2%
c5315	sizing	3,57
	buff	3,20
	gain	12%
c6288	sizing	7,98
	buff	7,74
	gain	3%
c7552	sizing	3,08
	buff	2,60
	gain	18%

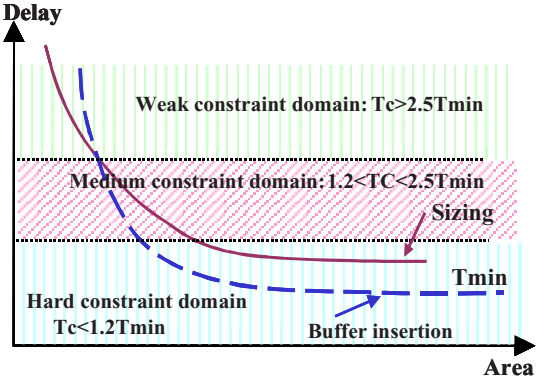

Fig. 6. Constraint domain definition.

Library characterization (Flimit determination)

Characterization of the optimization space

- Path classification
- Delay bounds determination: T_{max}, T_{min}

Delay constraint, Tc, distribution

- Delay constraint < minimum delay ➡ Buffer insertion
- Delay constraint > minimum delay
  - Weak constraint (Tc > 2.5Tmin) ➡ Gate sizing
  - Medium constraint (1.2 Tmin < Tc < 2.5 Tmin) ➡ Buffer insertion
  - Hard constraint (Tc < 1.2 Tmin) ➡ Buffer insertion & Global sizing

Fig. 7. Optimization protocol

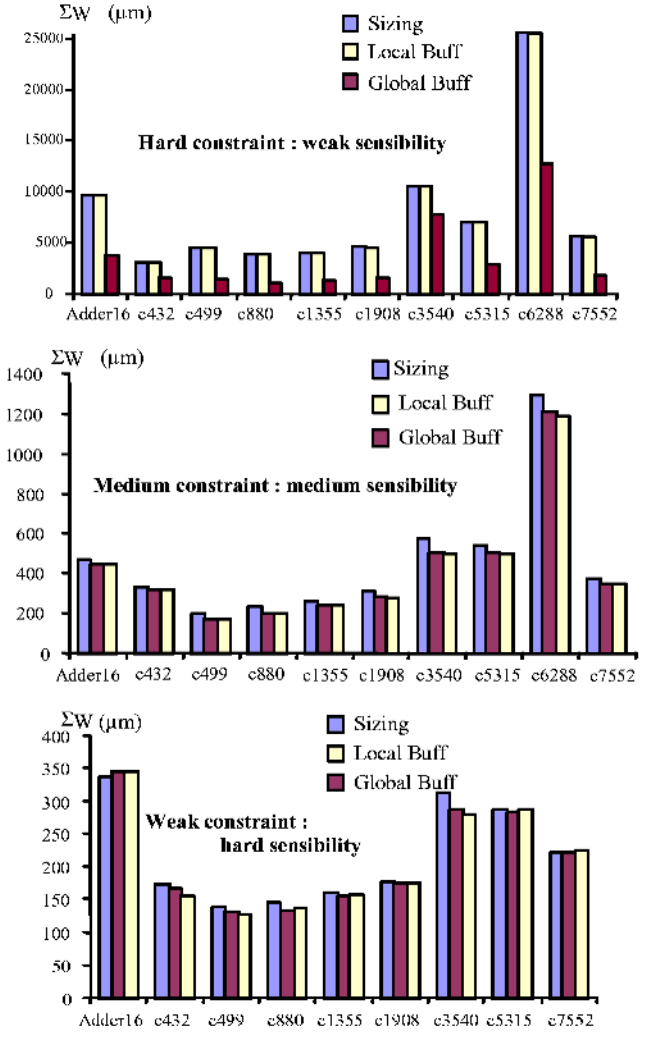

Fig. 8. Area saving in the different constraint domains for different optimization methods.

most efficient alternative. Note that these conditions defined with respect to the lower bound are circuit independent.

The resulting optimization protocol, (Fig.7), has been implemented in POPS for validation on the different ISCAS benchmarks. The comparison of the different steps is illustrated in Fig.8 where for three different delay constraint values (weak, medium, hard) we compare the path implementation area on the ISCAS circuits.

As shown, if for weak and medium constraints the different optimization methods are quite equivalent in terms of area, for hard constraint the buffer insertion with global sizing always results in important area saving.

5 Conclusion

Based on a realistic model for gate timing performance, we have defined metrics for selecting path optimization alternatives. We have proposed a method for determining the minimum delay, Tmin, achievable on a path. Then we have defined, at gate level, the fan out limit for buffer insertion, Flimit. Flimit has been used to determine the path critical nodes and Tmin, to select between sizing and buffer insertion alternatives. We have defined a gate sensitivity factor "a", to distribute the delay constraint, allowing path optimization at provably minimum area cost. These metrics have been used to define a general path optimization protocol that has been implemented in an optimization tool.

Validation on various benchmark circuits has demonstrated the validity of the defined boundaries for selecting between the different optimization alternatives.

References

[1] J. M. Shyu, A. Sangiovanni-Vincentelli, J. Fishburn, A. Dunlop, "Optimization-based transistor sizing" IEEE J. Solid State Circuits, vol.23, n°2, pp.400-409, 1988.

[2] J.P. Fishburn, A.E. Dunlop, "Tilos : a posynomial programming approach to transistor sizing", Proc. IEEE Int. Conf. Computer-Aided Design, pp. 326-328, 1985.

[3] S.S. Sapatnekar, V.B. Rao, P.M. Vaidya, S.M. Kang, "An exact solution to the transistor sizing problem for CMOS circuits using convex functions", IEEE trans. CAD, pp. 1621-1634, November 1993.

[4] I.E. Sutherland, B. Sproull, D. Harris, "Logical effort, designing fast cmos circuits", Morgan Kaufmann Publishers, Inc., 1999.

[5] S.R. Vemuru, A.R. Thorbjornsen, A.A. Tuszynski, " CMOS tapered buffer", IEEE J. Solid State Circuits, vol.26, n°9, pp.1265-1269,1991.

[6] Y. Jiang, S.S. Sapatnekar, C. Bamji, J. Kim, "Interleaving buffer insertion and transistor sizing into a single optimization", IEEE Trans. on VLSI, pp. 625-633, December 1998.

[7] P.G. Paulin, F. J. Poirot, "Logic decomposition algorithm for the timing optimization of multilevel logic", Proc. ICCD 89, pp.329-333.

[8] D. Singh, J.M. Rabaey, M. Pedram, F. Catthoor, S. Rajgopal, N. Sehgal, T.J. Mozdzen, "Power Conscious CAD tools and Methodologies: a Perspective", Proc. IEEE, vol.83, n°4, pp.570-593, April 1995.

[9] H.C. Chen, D.H.C. Du and L.R. Liu, "Critical Path Selection for Performance Optimization", IEEE trans. On CAD of Integrated Circuits and Systems, vol. 12, n°2, pp. 185-195, February 1995

[10] N. Azemard, D. Auvergne, "POPS : A tool for delay/power performance optimization ", Journal of Systems Architecture, Elsevier, n°47, pp375-382, 2001.

[11] S. Yen, D. Du and S. Ghanta,, "Efficient Algorithms for Extracting the k Most Critical Paths in Timing Analysis", Design Automation Conference, pp. 649-654, June 1989.

[12] S. Cremoux, N. Azemard, D. Auvergne, "Path resizing based on incremental technique", Proc. ISCAS, Monterey, USA, 1998.

[13] K.O. Jeppson, “Modeling the Influence of the Transistor Gain Ratio and the Input-to-Output Coupling Capacitance on the CMOS Inverter Delay”, IEEE JSSC, Vol. 29, pp. 646-654, 1994.

[14] P. Maurine, M. Rezzoug, N. Azémard, D. Auvergne “Transition time modeling in deep submicron CMOS” IEEE trans. on Computer Aided Design, Vol.21, n°11, pp.1352-1363, nov. 2002.

[15] Mead, M. Rem, "Minimum propagation delays in VLSI", ", IEEE J. Solid State Circuits, vol.SC17, n°4, pp.773-775, 1982.